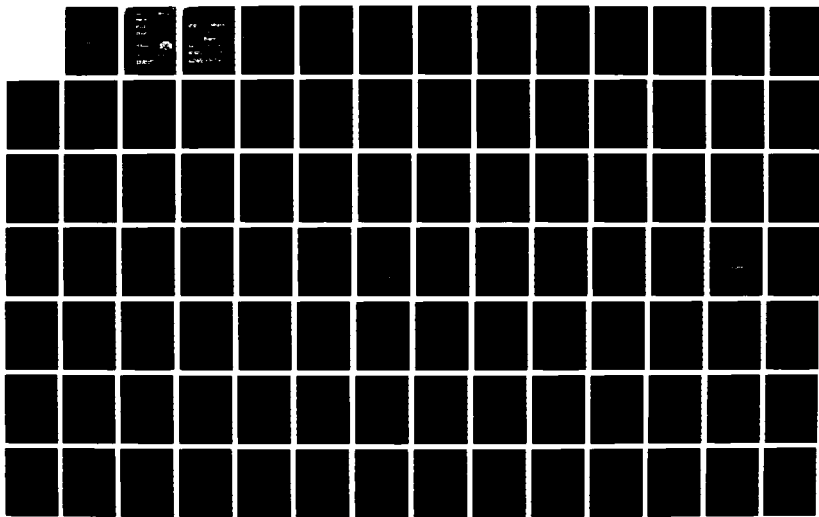
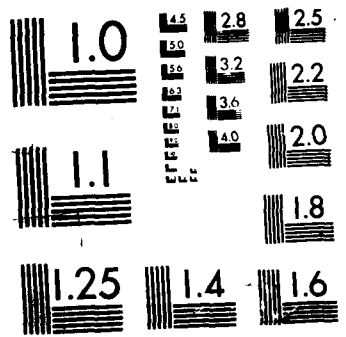


AD-A194 244 MINICOMPUTER AND DATA ANALYSIS IN SUPPORT OF THE AFGL 1/2  
(AIR FORCE GEOPHYSI.. (U) BOSTON UNIV MA  
R B D'AGOSTINO ET AL. 22 OCT 84 AFGL-TR-84-0276  
UNCLASSIFIED F19628-82-K-0044 F/Q 14/2 NL





AD-A194 244

AFGL-TR-84-0276

DTIC FILE COPY

**Minicomputer and Data Analysis in Support of the AFGL  
Radar Network**

Ralph B. D'Agostino  
Austin F.S. Lee  
Mary Ann Lockley

Trustees of Boston University  
881 Commonwealth Avenue  
Boston, MA 02215

October 32, 1984

Final Report  
19 May 1982-30 September 1983

DTIC  
ELECTE  
JUN 13 1985  
S D  
G H

APPROVED FOR PUBLIC RELEASE, DISTRIBUTION UNLIMITED

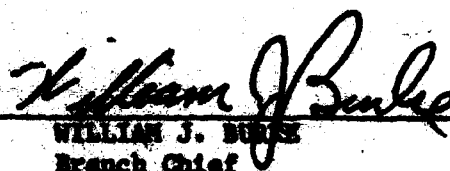
AIR FORCE GEOPHYSICS LABORATORY  
AIR FORCE SYSTEMS COMMAND  
U.S. AIR FORCE  
HANSCOM AIR FORCE BASE, MA 01931

8 6 8 030

"This technical report has been reviewed and is approved for publication"



DAVID J. KNECHT  
Contract Manager



WILLIAM J. BURKE  
Branch Chief

FOR THE COMMANDER

  
RITA C. SAGALY  
Division Director

This report has been reviewed by the ESD Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS).

Qualified requesters may obtain additional copies from the Defense Technical Information Center. All others should apply to the National Technical Information Service.

If your address has changed, or if you wish to be removed from the mailing list, or if the addressee is no longer employed by your organization, please notify AFGL/DAA, Hanscom AFB, MA 01731. This will assist us in maintaining a current mailing list.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

A194244

## REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION <b>unclassified</b>			1b. RESTRICTIVE MARKINGS										
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release, distribution unlimited										
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			5. MONITORING ORGANIZATION REPORT NUMBER(S) AFGL-TR-84-0276										
4. PERFORMING ORGANIZATION REPORT NUMBER(S)			5. MONITORING ORGANIZATION REPORT NUMBER(S) AFGL-TR-84-0276										
6a. NAME OF PERFORMING ORGANIZATION Boston University		6b. OFFICE SYMBOL (If applicable)		7a. NAME OF MONITORING ORGANIZATION Air Force Geophysics Laboratory									
6c. ADDRESS (City, State and ZIP Code) 881 Commonwealth Avenue Boston, MA 02215		7b. ADDRESS (City, State and ZIP Code) Hanscom Air Force Base, MA 01731											
8a. NAME OF FUNDING/SPONSORING ORGANIZATION		8b. OFFICE SYMBOL (If applicable)		9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F19628-82-K-0044									
8c. ADDRESS (City, State and ZIP Code)		10. SOURCE OF FUNDING NOS.											
		<table border="1"><tr><td>PROGRAM ELEMENT NO.</td><td>PROJECT NO.</td><td>TASK NO.</td><td>WORK UNIT NO.</td></tr><tr><td>62101F</td><td>7601</td><td>08</td><td>AT</td></tr></table>				PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.	62101F	7601	08	AT
PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.										
62101F	7601	08	AT										
11. TITLE (Include Security Classification) Mini computer and Data Analysis in Support of the AFGL Magnetometer Network													
12. PERSONAL AUTHOR(S) D'Agostino, Ralph B., Lee, Austin F.S., Leekley, Mary Ann													
13a. TYPE OF REPORT final		13b. TIME COVERED FROM 5/19/82 TO 9/30/83		14. DATE OF REPORT (Yr., Mo., Day) 84/10/22									
				15. PAGE COUNT 110									
16. SUPPLEMENTARY NOTATION													
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)										
FIELD	GROUP	SUB. GR.	Operating system, Magnetometer network, filtering, spectral analysis										
19. ABSTRACT (Continue on reverse if necessary and identify by block number)													
Implementation of a new operating system, including hardware and software for the magnetometer network was discussed. A computer package for filtering and spectral analysis was described, including flow charts, user's directory, mathematical foundations and examples.													
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED <input checked="" type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS <input type="checkbox"/>			21. ABSTRACT SECURITY CLASSIFICATION unclassified										
22a. NAME OF RESPONSIBLE INDIVIDUAL David Knecht			22b. TELEPHONE NUMBER (Include Area Code) 617/377-3240		22c. OFFICE SYMBOL AFGL/PHG								

## CONTENTS

1. Description of Filtering/Spectral Analysis Programs (I) .....	1
2. Description of Filtering/Spectral Analysis Programs (II) .....	9
3. Description of Filtering/Spectral Analysis Programs (III) .....	16
4. Description of Filtering/Spectral Analysis Programs (IV) .....	50
5. Implementing Vortex II .....	76
Project Origins .....	78
Specification Changes .....	81
System Generation .....	82
Software Modifications .....	88
Specialized Software .....	90
System Present and Future .....	93
6. Bibliography .....	95
7. Appendix A: AFGL Vortex II User Guide .....	97



<b>Accession For</b>	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
<b>Availability Codes</b>	
Dist	Avail and/or Special
A-1	

## **Description of Filtering/Spectral Analysis Programs**

by

Austin F.S. Lee, Ph.D.  
and  
Ralph B. D'Agostino, Ph.D.

The following describes the computer package originally developed by Dr. Paul F. Fougere. The program performs data input, data cleaning and filtering, spectral analysis, and finally graphical output.

### **OUTLINE**

In the first section, all subprograms that are used in this package are listed according to the sequence they appear in the deck. In the second section, the purposes of some important subprograms are described. Finally, the structure of the entire package and the interrelations among those subprograms are presented in the form of a chart.

## I. List of Subprograms

1. PROGRAM CALL - OVERLAY (MAIN, 0, 0)
2. GAUSS
3. NLOGN
4. SCALE
5. AXIS
6. PLOT
7. PROGRAM REMEZ1 - OVERLAY (MAIN, 1, 0)
8. SUMMER
9. EFF
10. WATE
11. ERROR
12. REME Z
13. D
14. GEE
15. OUCH
16. FQRESP
17. PROGRAM FAST - OVERLAY (MAIN, 2, 0)
18. SET3
19. DATA
20. INPU
21. NORM
22. SETUP2
23. SETFIL
24. PROGRAM SPECTRA - OVERLAY (MAIN, 3, 0)
25. NORM
26. QUAD
27. BPEC
28. SPEC
29. INTEG
30. PEAK
31. WINTER
32. SPRING
33. INP
34. LOGIC
35. PRINTER
36. RUNAV
37. HIPASS

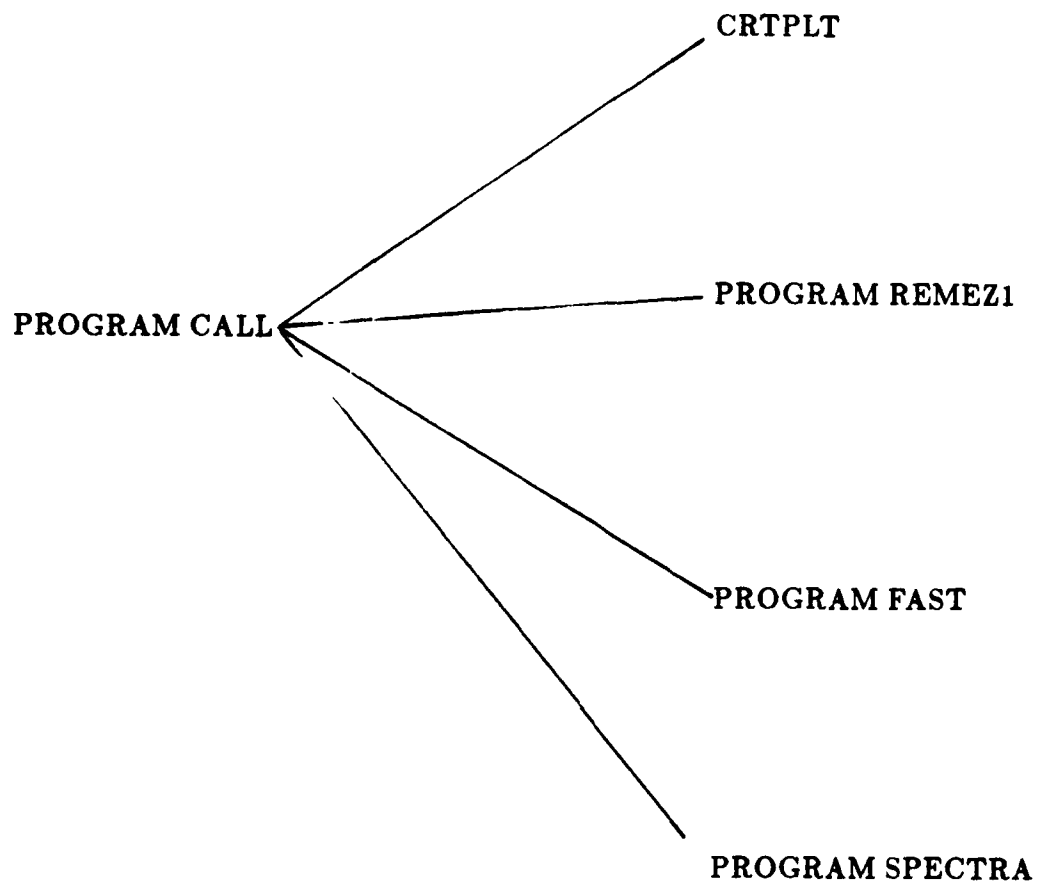


## II. Purposes of Some Important Subprograms

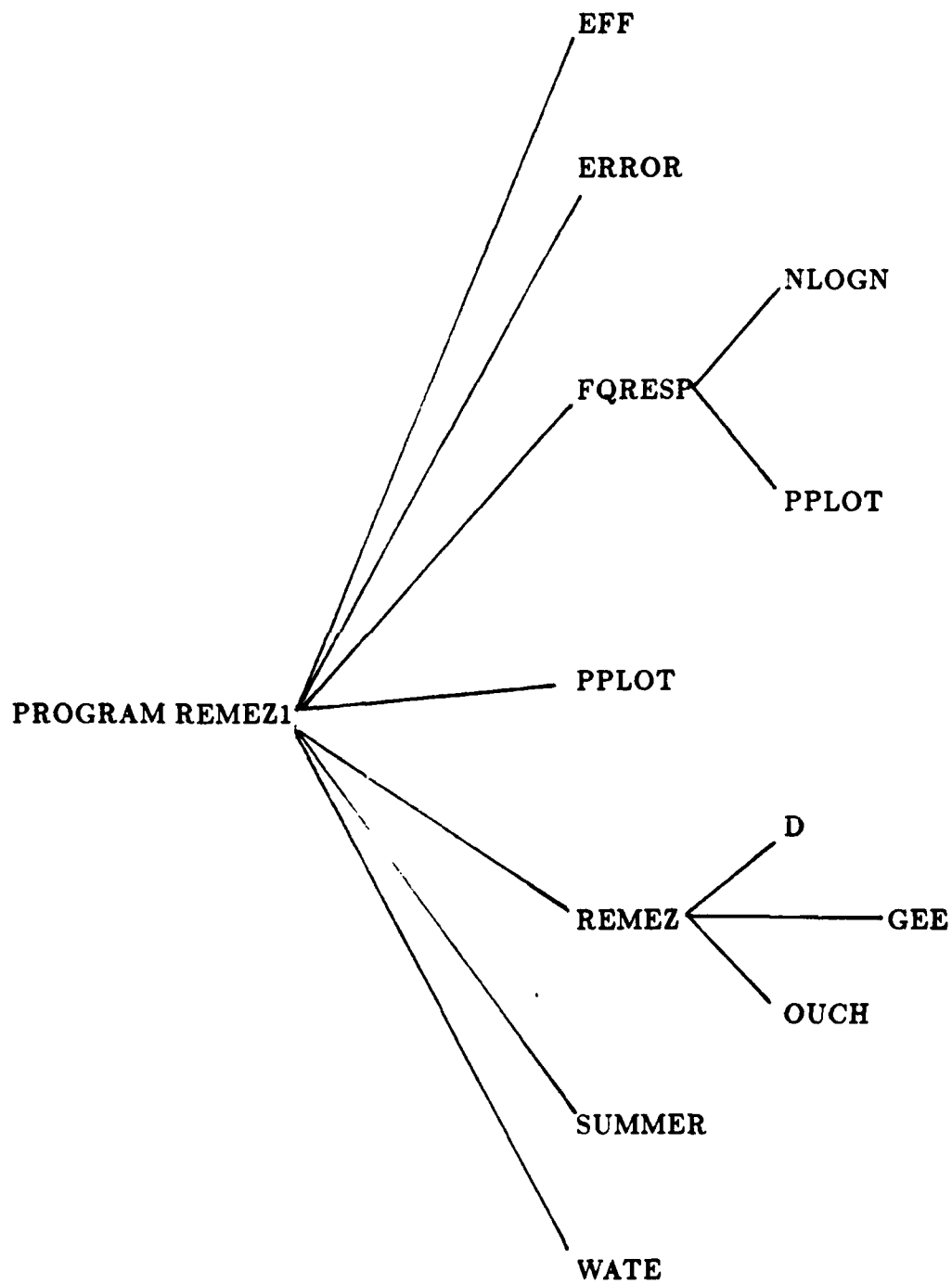
1. GAUSS: normal random number generator.
6. PPLOT: plotting subroutine for calcamp plotter.
7. PROGRAM REMEZ1: program for the design of linear phase finite impulse response filters using the Remez exchange algorithm.
9. EFF: to calculate the desired magnitude response as a function of frequency.
10. WATE: to calculate the weight function as a function of frequency.
12. REMEZ: to implement the Remez exchange algorithm for the weighted Chebychev approximation of a continuous function with a sum of cosines.
13. D: to calculate the Lagrange interpolation coefficients for use in the function GEE.
14. GEE: to calculate the frequency response using the Lagrange interpolation formula in the Barycentric form.
15. OUCH: to flag "failure to converge."
24. PROGRAM SPECTRA: to estimate the power spectrum of a discrete time series or of a sampled continuous process, using the maximum entropy technique of J.R. Burg.
26. QUAD: to perform quadratic interpolation.
29. INTEG: integration routine using the Simpson's rule.
30. PEAK: this subroutine finds all spectral peaks which are larger than 1% of the largest original spectral height. Replaces each original spectral height with integral under elementary spectral cell divided by size of cell. Uses repeated interpolation to locate each peak precisely, such that three points which bracket peak all lie within 1% of amplitude at peak.
31. WINTER: to find area under spectrum between two given points.
33. INP: a sample of the input routine which must be written by the user.
36. RUNAV: to calculate running average.

III. Structure of the Package and Interrelations of All Subprograms (Subprograms that are not supplied by the package are underlined)

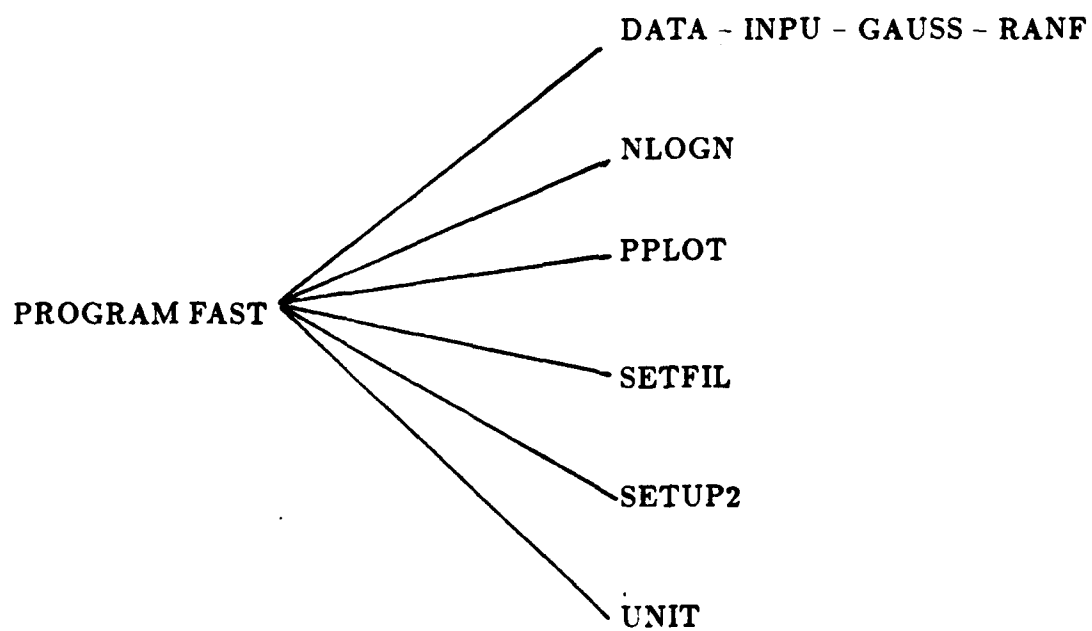
Stage 1:



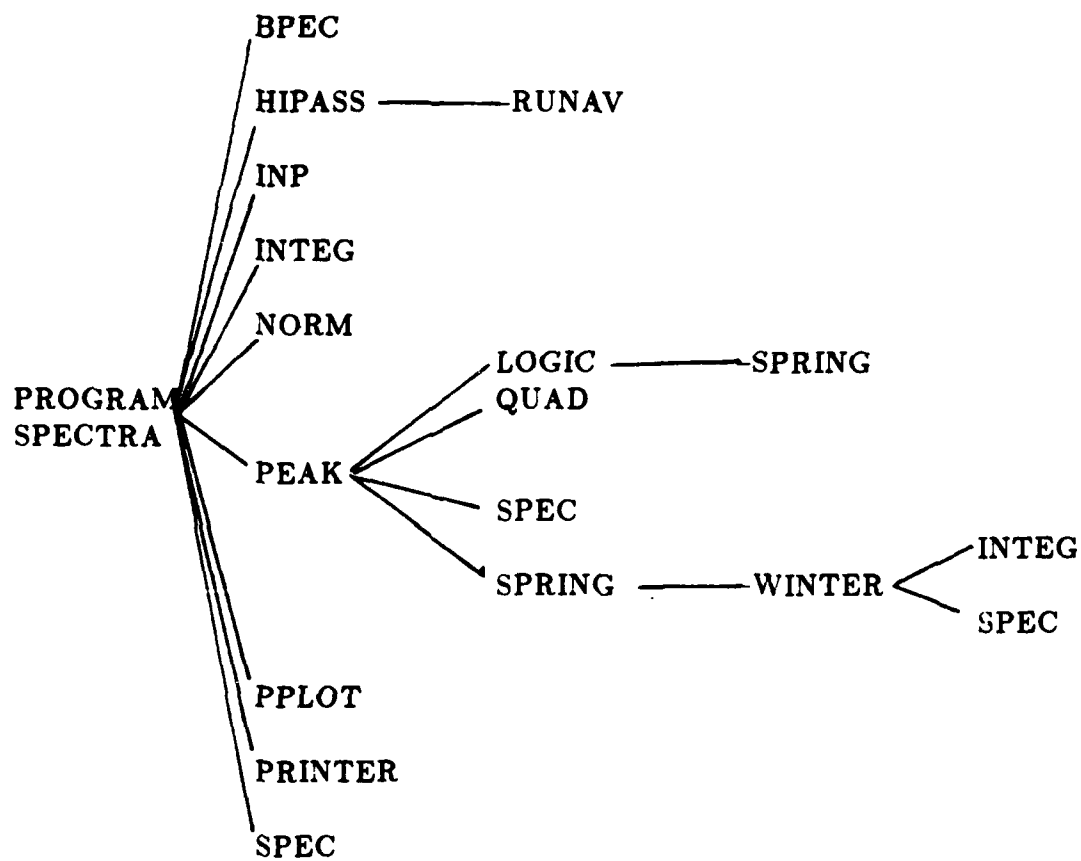
Stage 2:



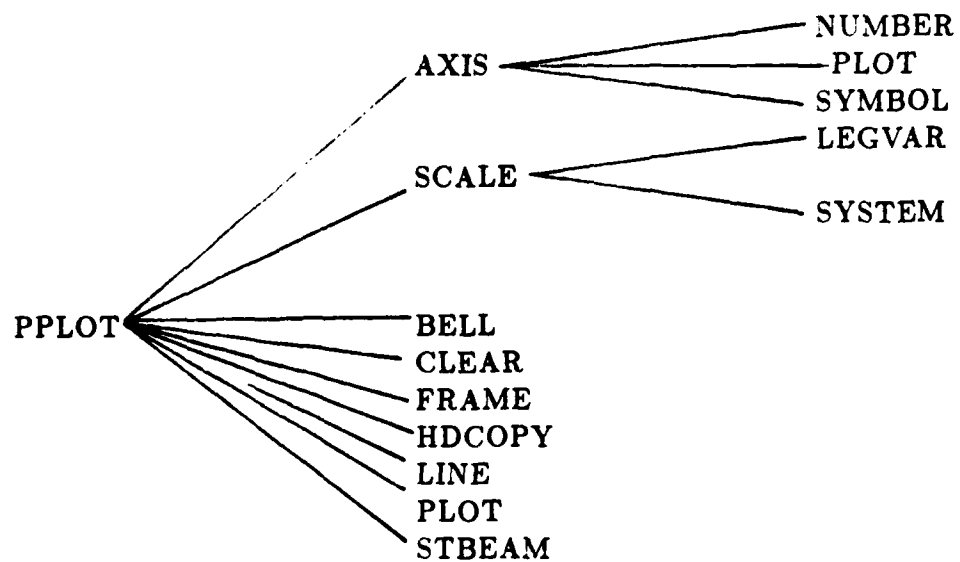
Stage 3:



Stage 4:



"PLOT" that appears several times in Stages 2, 3, and 4 follows the following structure.



## **Description of Filtering/Spectral Analysis Program (II)**

by

Austin F.S. Lee, Ph.D.

and

Ralph B. D'Agostino, Ph.D.

This is the second part of the document that describes the filtering/spectral analysis programs originally developed by Dr. Paul F. Fougere. The interactive data input sequential structure of the program is fully described here. Any user who intends to use the programs for filtering and spectral analysis may follow the procedures step by step as presented in this report.

## Interactive Data Input Sequential Structure

### A. Main Program

1. "Type I1, I2, I3, if subroutine(s) is (are) to be called."

Example: 0 1 0

Explanation:

I1 = 0, to call REMEZ1;

I2 = 0, to call FAST;

I3 = 0, to call SPCTRA.

2. "Type Kplot, sf."

Example: 1 ?

Explanation:

Kplot > 0 to call CRTPLT (IP, sf, 17.);

sf = parameter in CRTPLT.

### B. Subroutine REMEZI

1. "Enter filter length. Max  $\leq 358$ ."

Example: 32

Explanation

nfalt=filter < 358, then

(1) if nfalt < -999, then

a. read nfilt, nfcns, ( $h(I)$ ,  $I = 1, nfcns$ ) from unit 4.

call subroutine SUMMER

subroutine SUMMER "If you want to modify sum, enter new sum." If  
new sum < 0, no modification is made."

Example: -1 Explanation:

Bum  $\geq 0$  to modify sum.

c. finally, go to step 10 below.

- (2). If  $-999 \leq \text{nfalt} \leq 0$ , then return to main program.

2. "Filter type ? 1. Bandpass, 2. Differentiators,

3. Hibert Transform filter."

Example: 1

Explanation:

Jtype-type of filter  $\left\{ \begin{array}{l} 1. = \text{Bandpass filter;} \\ 2. = \text{Differentiators;} \\ 3. = \text{Hibert Transform filter.} \end{array} \right.$



3. "Number of bands?"

Example: 3

Explanation:

Nbands-number of bands.

4. "Enter band edges."

Example: 0 0.0 0.2 0.35 0.425 0.5

Explanation:

edge ( $J$ ),  $J = 1 \dots 2 * \text{n-bands}$ -band edges

5. "Enter desired value for each band."

Example: 0 1 0

Explanation:

$fx(J)$ ,  $J = 1 \dots \text{nbands}$ -desired value for each band.

6. "Enter weight for each band."

Example: 10 1 10

Explanation:

$wtx(J)$ ,  $J = 1 \dots \text{nbands}$ -weight for each band.

7. "Enter additive constant."

Example: 1

Explanation:

fadd additive constant.

8. "Enter A and B."

Example: 1 1

Explanation:

A=multiplier; B=exponent, used in a formula in

Function EFF; i.e.,  $EFF = A * \text{temp} ** B$ .

9. "Enter 1 to plot desired response."

Example: 1

Explanation:

key = 1 to call PLOT to plot desired response.

10. "print impulse response? enter 1."

Example: 1

Explanation:

key = 1 to print impulse response.

11. "Frequency response ? Enter 1."

Example: 1

Explanation:

key = 1, to call FQRESP to compute frequency response.

### C. Subroutine FQRESP

1. "Linear plot? Enter 1."

Example: 1

Explanation:

key = 1, to call PLOT for linear plot.

2. "For logarithmic X axis, enter 1."

Example: 1

Explanation:

Log = 1, to use logarithmic X axis in PLOT.

3. "Plot in DB ? Enter 1."

Example: 1

Explanation:

key = 1, to plot in DB.

4. "Zoom ? Enter 1."

Example: 1

Explanation:

key = 1, zoom.

5. "Enter 2 frequencies in range 0, 0.5."

Example: 0.1 0.4

Explanation:

two frequencies in range (0, 0.5)

$$K_{bot} = 1 + \text{int} (2048 * f1)$$

$$K_{top} = 1 + \text{int} (2048 * f2)$$

### D. Subroutine FAST

1. "Read in filter ?"

"If no, type number  $\leq 0$ ,"

"If yes, type number of filter weights."

Example: 0

Explanation:

$$\text{namm} \begin{cases} \leq 0, & \text{no filter} \\ > 0, & \text{number of filter weights.} \end{cases}$$

2. "Use Tape 4 ? Enter 1."

Example: 1

Explanation:

key = 1, use unit 4 to read nfil, nfile,

$(x(I), I = 1, n \text{ fil } 2)$

$X(I)$  are filter coefficients.

3. Read nfil, nfile,  $(x(I), I = 1, n \text{ file})$  from unit 4,

then skip to step 5 below.

4. "Type in filter coefficients."

Example: Nfile real numbers

Explanation:

$x(I), I = 1, n \text{ file}$ —filter coefficients.

5. "Type number of data points."

Example: 1 2 0

Explanation:

Number of data points.

6. "Plot input data ? Enter 1."

Example: 1

Explanation:

key = 1, to call PPLOT to plot input data.

7. "Plot filtered data? Enter 1."

Example: 0

Explanation:

key = 1, to call PPLOT to plot filtered data.

#### E. Subroutine SPCTRA

1. "Type Istart, istip, inc. Returns to main program if  $i \text{start} \leq 0$ ."

Example: 1 4 3

Explanation:

if  $i \text{start} \leq 0$ , no spectral analysis.

2. "Enter DT (sampling time interval) and two frequencies in range  $(0, 0.5 * DT)$ ."

Example: 1 0 0.5

Explanation:

DT - sampling time interval;

F1, F2 - two frequencies in the range  $(0, 0.5 * DT)$ .

3. "Enter any 10 character name for frequency axis."

Example: freqaxis

Explanation:

Name - name for frequency axis, maximum 10 characters.

4. "For logarithmic X axis, enter 1."

Example: 1

Explanation:

Log = 1, for logarithmic x axis.

5. "Type number of observations, if  $\leq 0$ , then use previous data."

Example: 0

Explanation:

key = number of observations, if key  $\leq 0$ , then use previous data.

6. "Remove running mean from data? Enter number of points (or 0)."

Example: 0

Explanation:

nhipas: number of points for computing moving average to detrend data.

7. "plot input data ? Enter 1."

Example: 1

Explanation

key = 1, to call PLOT to plot input data.

8. "normalize data ? Enter 1."

Example: 1

Explanation:

key = 1, to normalize data.

9. "Also remove linear trend ? 1 = yes, 0 = No."

Example: 0

Explanation:

$$mey = \begin{cases} 1 & \text{to remove linear trend;} \\ 0 & \text{not to remove linear trend.} \end{cases}$$

10. "Plot normalized data ? Enter 1."

Example: 1

Explanation:

key = 1 to plot normalized data.

11. "Plot PSD ? (0=linear PSD, 1=Log PSP, -1=no plot)" "If you want both linear and log, start with 0 (linear)"

Example: 1

Explanation:

$$\text{John} = \begin{cases} 0, & \text{to plot linear PSD;} \\ 1, & \text{to plot log PSD;} \\ -1, & \text{no plot, go back to Step 1 in this subroutine.} \end{cases}$$

## **Description of Filtering/Spectral Analysis Program (III)**

by

**Austin F.S. Lee, Ph.D.**

and

**Ralph B. D'Agostino, Ph.D.**

This is the third part of the document that describes the filtering/spectral analysis program. This analytic system constitutes two major portions (see Part I of the report):

- (1) **REMEZ1**: To design linear phase finite impulse response filters using the Remez exchange algorithm for the weighted Chebychev approximation of a continuous function with a sum of cosines.
- (2) **SPECTRA**: To estimate the power spectrum of a discrete time series or of a sampled continuous process using the maximum entropy technique of J.P. Burg.

In this part of the document, the mathematical background and the details of flow charts of REMEZ1 are provided. Some examples are also given to facilitate the applications of REMEZ1. The contents are as follows:

1. Mathematical background of REMEZ1
2. Flow charts
3. Examples

Three articles were referenced in preparation of this document, namely,

- (1) T.W. Parks and J.H. McClellan,  
"Chebyshev approximation for nonrecursive digital filters with linear phase," IEEE Trans. Circuit Theory, vol. CT-19, pp. 189-194, Mar. 1972.
- (2) J.H. McClellan, T.W. Parks, and L.R. Rabiner,  
"A computer program for designing optimal FIR linear phase digital filters," IEEE Trans. Audio Electroacoust., vol. AU-21, pp. 506-526, Dec. 1973.
- (3) L.R. Rabiner, J.H. McClellan, and T.W. Parks,  
"FIR digital filter design techniques using weighted Chebyshev approximation," Proc. IEEE, vol. 63, pp. 595-610, Apr. 1975.

## 1. Mathematical Background of REMEZ1

The frequency response of an FIR (finite impulse response) digital filter with an  $N$ -point impulse response  $\{h(k)\}$  is the  $z$ -transform of the sequence evaluated on the unit circle, i.e.,

$$H(f) = \sum_{k=0}^{N-1} h(k)e^{-j2\pi kf} \quad (1)$$

The frequency response of a linear phase filter can be written as

$$H(f) = G(f)e^{j\{ \frac{L\pi}{2} - (\frac{N-1}{2})2\pi f \}} \quad (2)$$

where  $G(f)$  is a real valued function,

$$L = \begin{cases} 0, & \text{if the impulse response is positively symmetric,} \\ 1, & \text{if the impulse response is negatively symmetric.} \end{cases}$$

Furthermore,  $G(f)$  can be expressed as

$$G(f) = Q(f)P(f) \quad (3)$$

where  $P(f)$  is a linear combination of cosine functions:

$$P(f) = \sum_{k=0}^{n-1} \alpha(k) \cos(2\pi kf) \quad (4)$$

Let  $D(f)$  be a desired magnitude response and  $W(f)$  be a positive weight function, then one wishes to find the optimal set of coefficients of  $G(f)$  which minimizes the maximum absolute weighted error (weighted Chebyshev error), defined as

$$\|E(f)\| = \max_{f \in F} W(f) |D(f) - G(f)| \quad (5)$$

where  $F$  is a compact subset of  $[0, \frac{1}{2}]$ . In the light of (3),  $\|E(f)\|$  in (5) can be rewritten as

$$\|E(f)\| = \max_{f \in F'} \hat{W}(f) |\hat{D}(f) - P(f)| \quad (6)$$

where

$$\begin{aligned} \hat{W}(f) &= W(f)Q(f) \\ \hat{D}(f) &= D(f)/Q(f) \end{aligned}$$

and

$$F' = F - \{ \text{endpoints where } Q(f) = 0 \}.$$

The following "alternation theorem" provides the set of necessary and sufficient conditions for the best approximation:

**Alternation Theorem:** If  $P(f)$  is a linear combination of  $r$  cosine functions, then a necessary and sufficient condition that  $P(f)$  be the unique best weighted Chebyshev approximation to a continuous function  $\hat{D}(f)$  on  $F'$  is that the weighted error function  $E(f) = \hat{W}(f)[\hat{D}(f) - P(f)]$  exhibits at least  $r + 1$  extremal frequencies in  $F$ .

These extremal frequencies are a set of points

$$\{F_i\}, i = 1, 2, \dots, r + 1 \text{ such that } F_1 < F_2 < \dots < F_r < F_{r+1}$$

with  $E(F_i) = -E(F_{i+1})$ ,  $i = 1, 2, \dots, r$  and

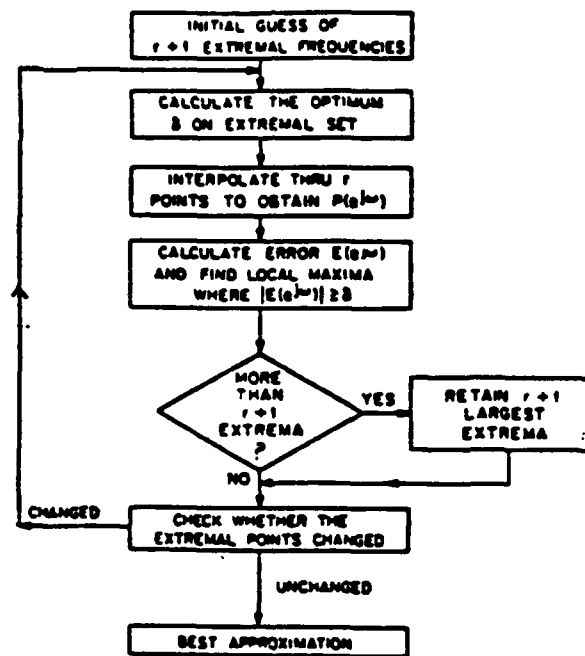
$$|E(F_i)| = \max_{f \in F'} E(f)$$

By applying the Alternation Theorem, the optimal set of coefficients  $\alpha(k)$ ,  $k = 0, 1, \dots, r - 1$ , can be found by solving the following set of equations:

$$\hat{W}(f_k)\{\hat{D}(f_k) - P(f_k)\} = (-1)^k \delta \quad (7)$$

where  $\delta$  is the absolute magnitude of the weighted error function at  $\{f_k\}$ . The Remez exchange algorithm is an algorithm which solves the Chebyshev approximation problem by searching for the extremal frequencies  $\{f_k\}$  of the best approximation. Figure 1 outlines the Remez exchange algorithm.





Flow chart of the Remez exchange algorithm.

Figure 1: Outline of the Remez exchange algorithm

## 2. Detailed flow charts of REMEZ 1 and Its Subroutines.

The following figures display the flow charts of programs contained in REMEZ 1.

Figure 2: Overall flow chart of filter design algorithm.

Figure 3: Outlined flow chart of REMEZ 1 (main program)

Figure 4: Detailed flow chart of REMEZ 1, showing all statements in the program.

Figure 5: Flow chart of Subroutine EFF.

Figure 6: Flow chart of Subroutine WATE.

Figure 7: Flow chart of REMEZ.

Figure 8: Flow chart for arbitrary magnitude filter design algorithm.

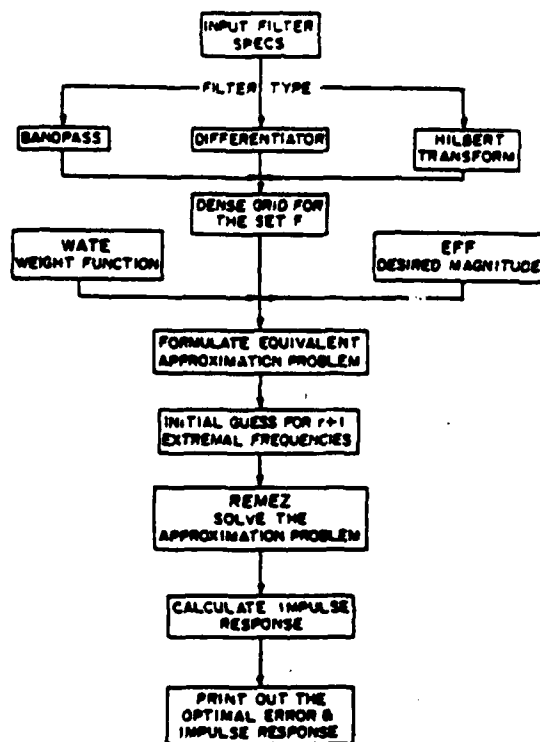


Figure 2: Overall flow chart of filter design algorithm

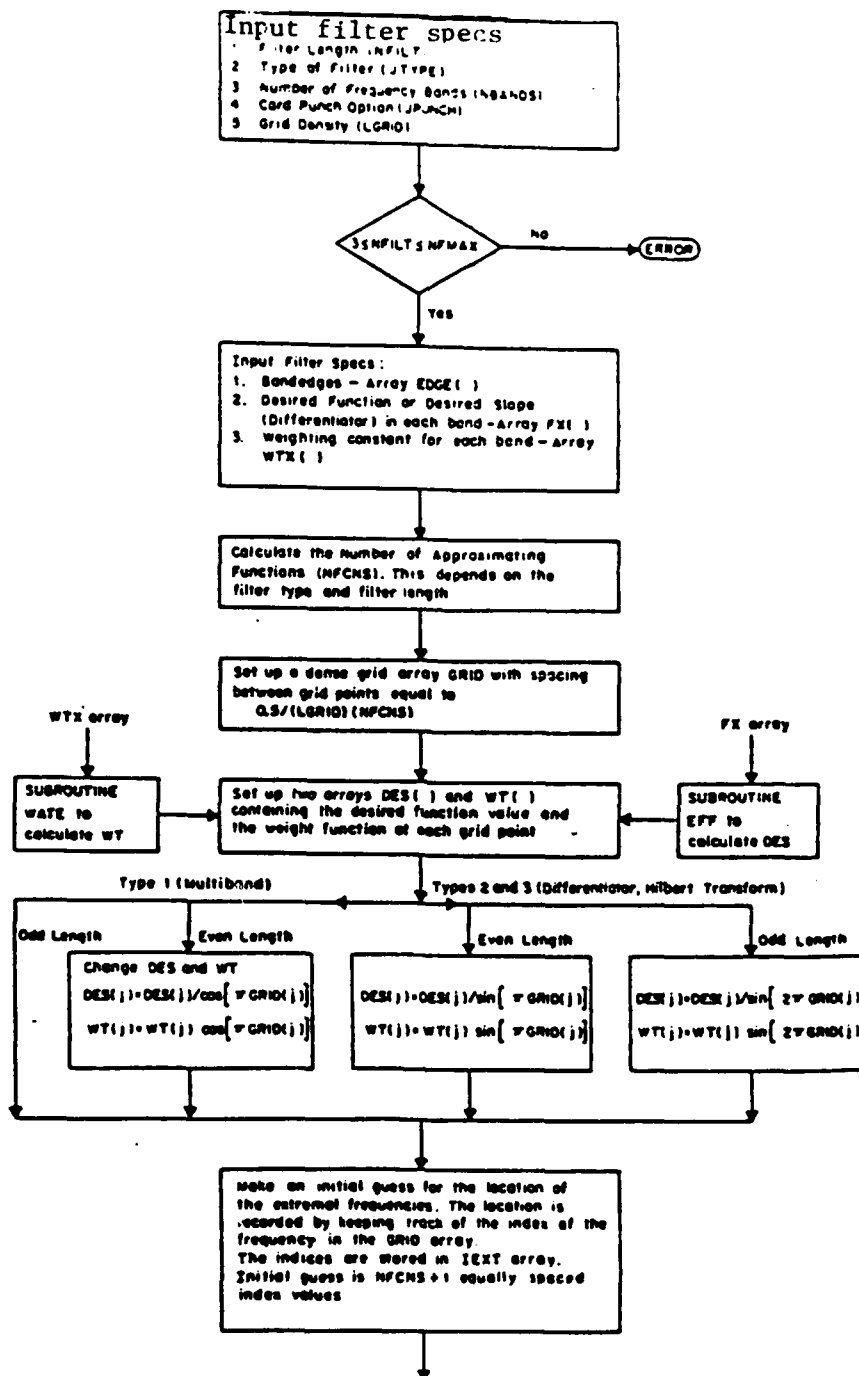


Figure 3: Outlined flow chart of REMEZ 1 (main program)

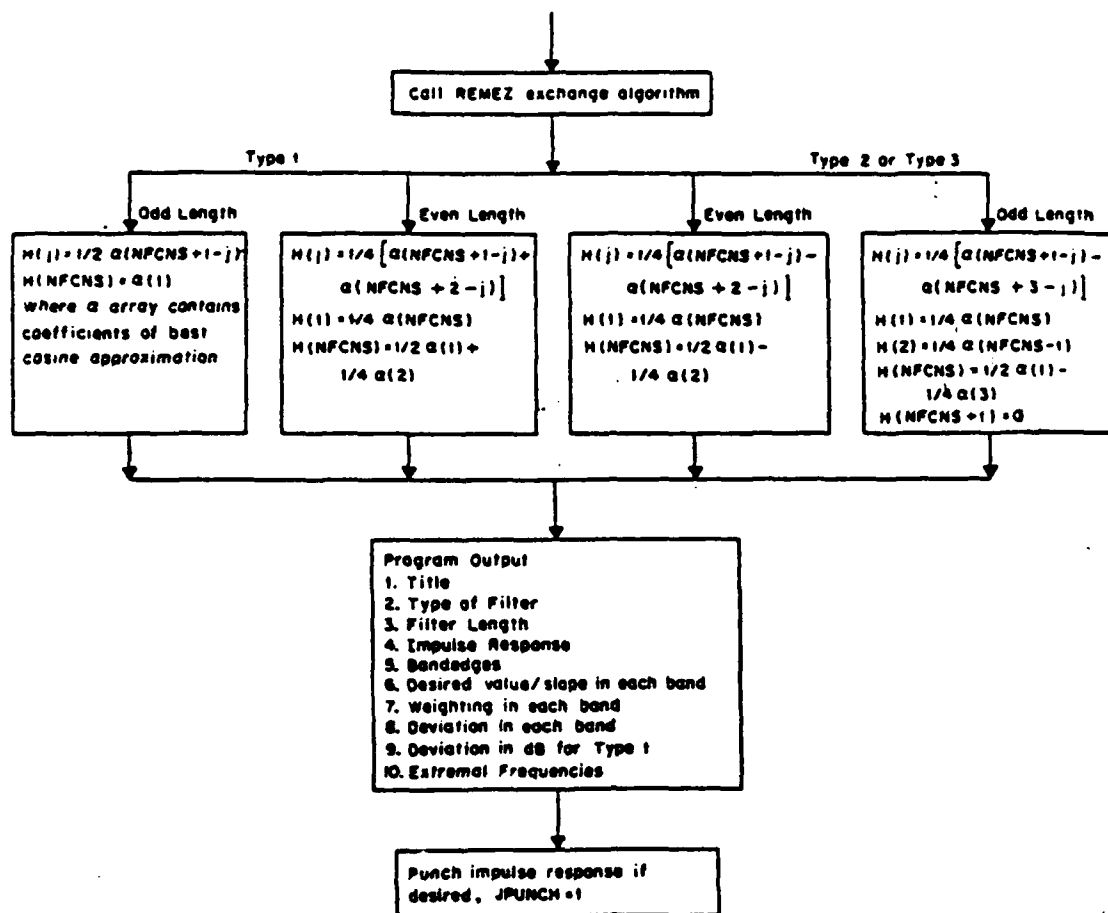


Figure 3: Outlined flow chart of REMEZ 1 (main program)- continued

Symbol shapes and their Meaning:



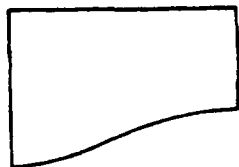
substitution statement



condition or IF statement



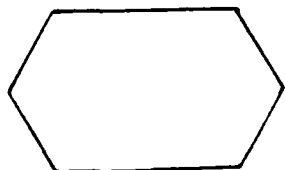
Input Statement



Output Statement



Unconditional transfer  
or GO TO Statement



Iteration or Do Statement



Statement label

Figure 4: Detailed flow chart of REMEZ 1, showing all statements in the program.

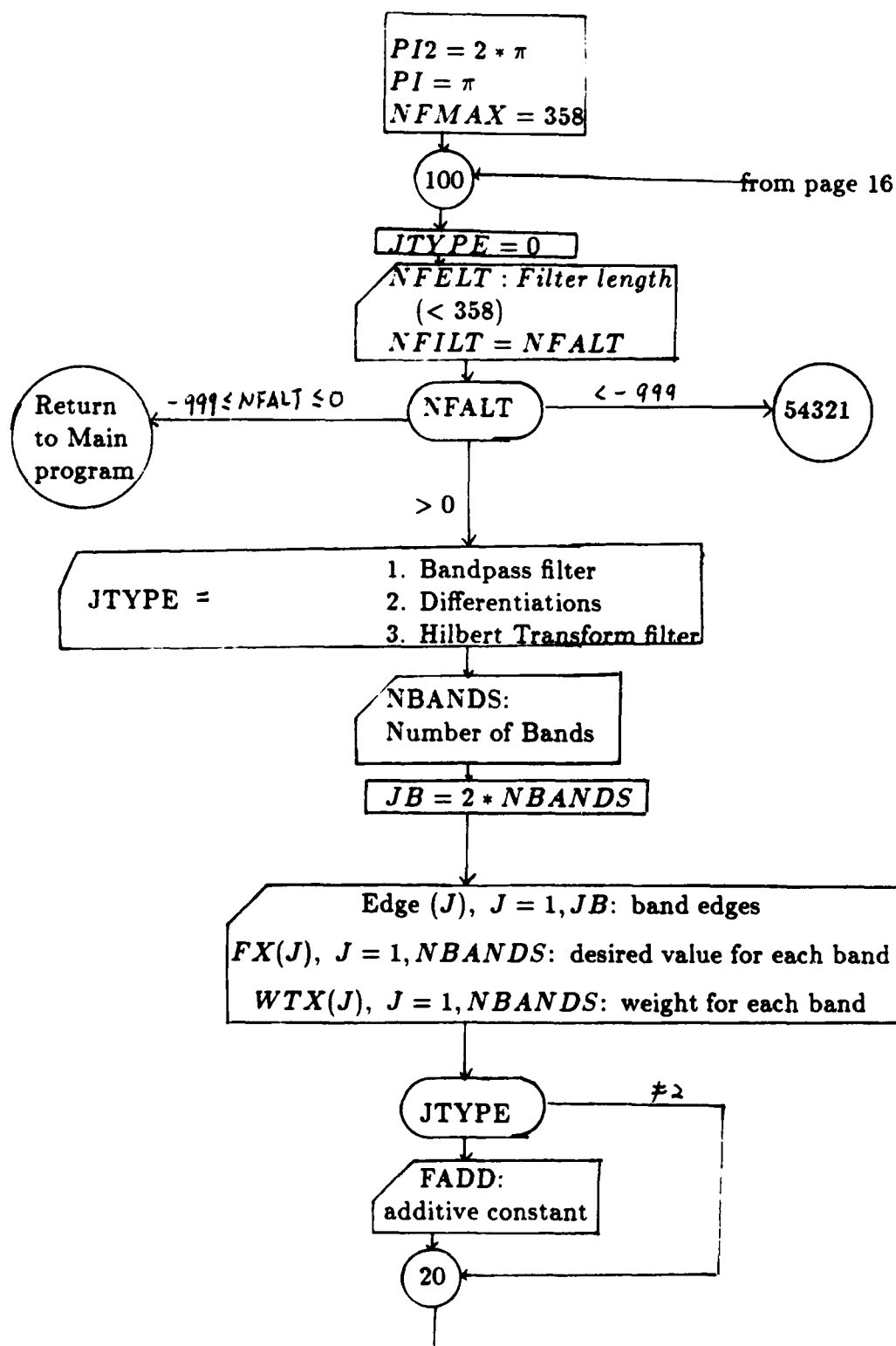
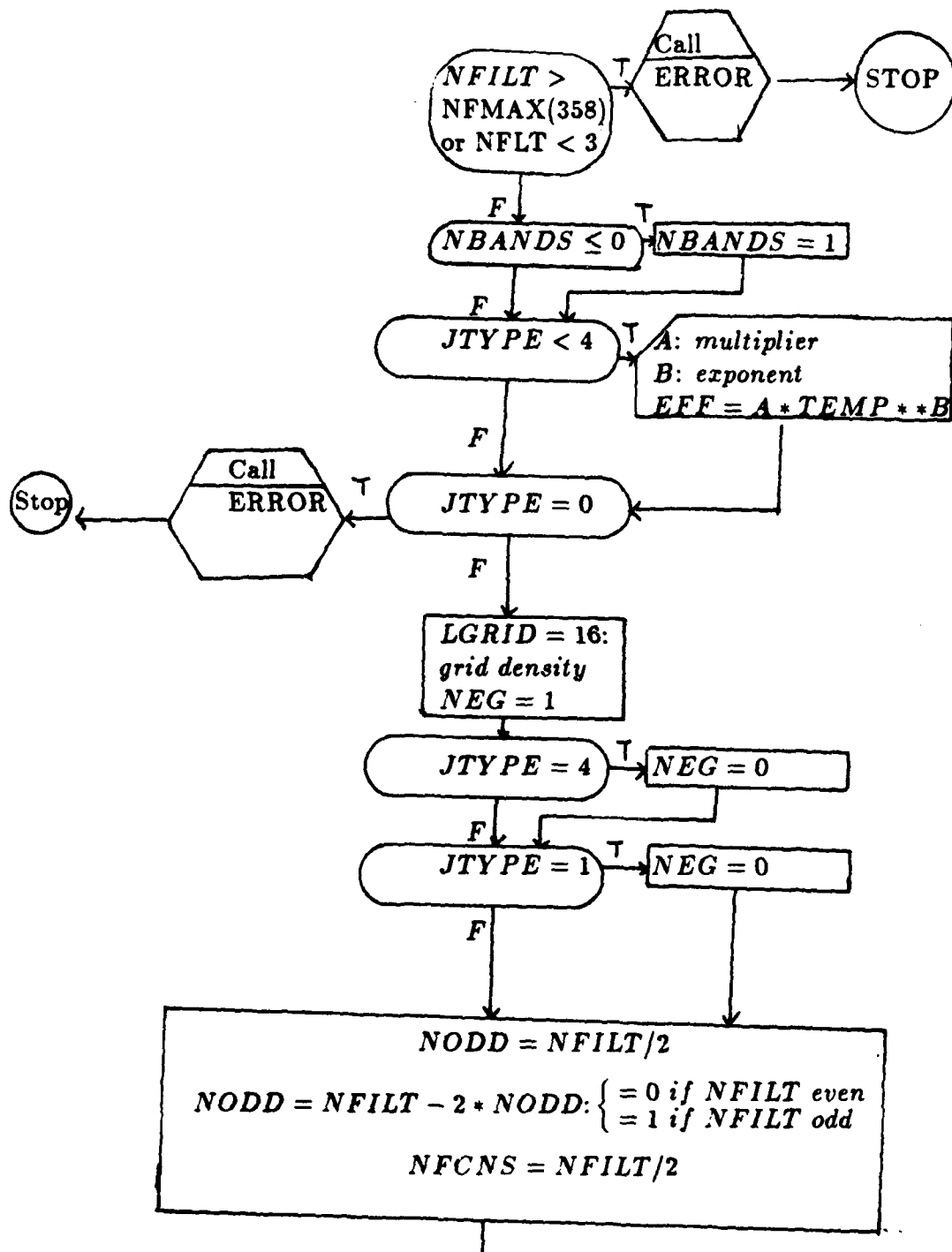


Figure 4: Detailed flow chart of REMEZ 1, showing all statements in the program - continued





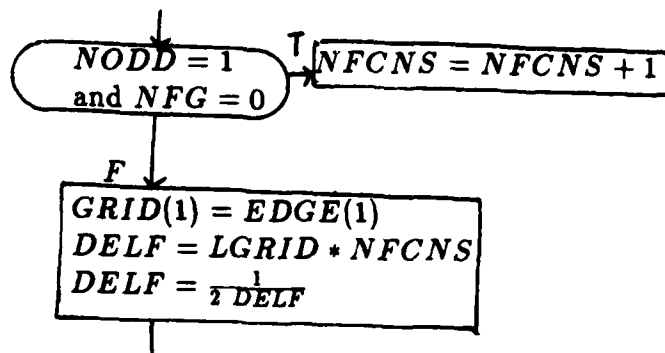


Figure 4: Detailed flow chart of REMEZ 1, showing all statements in the program continued

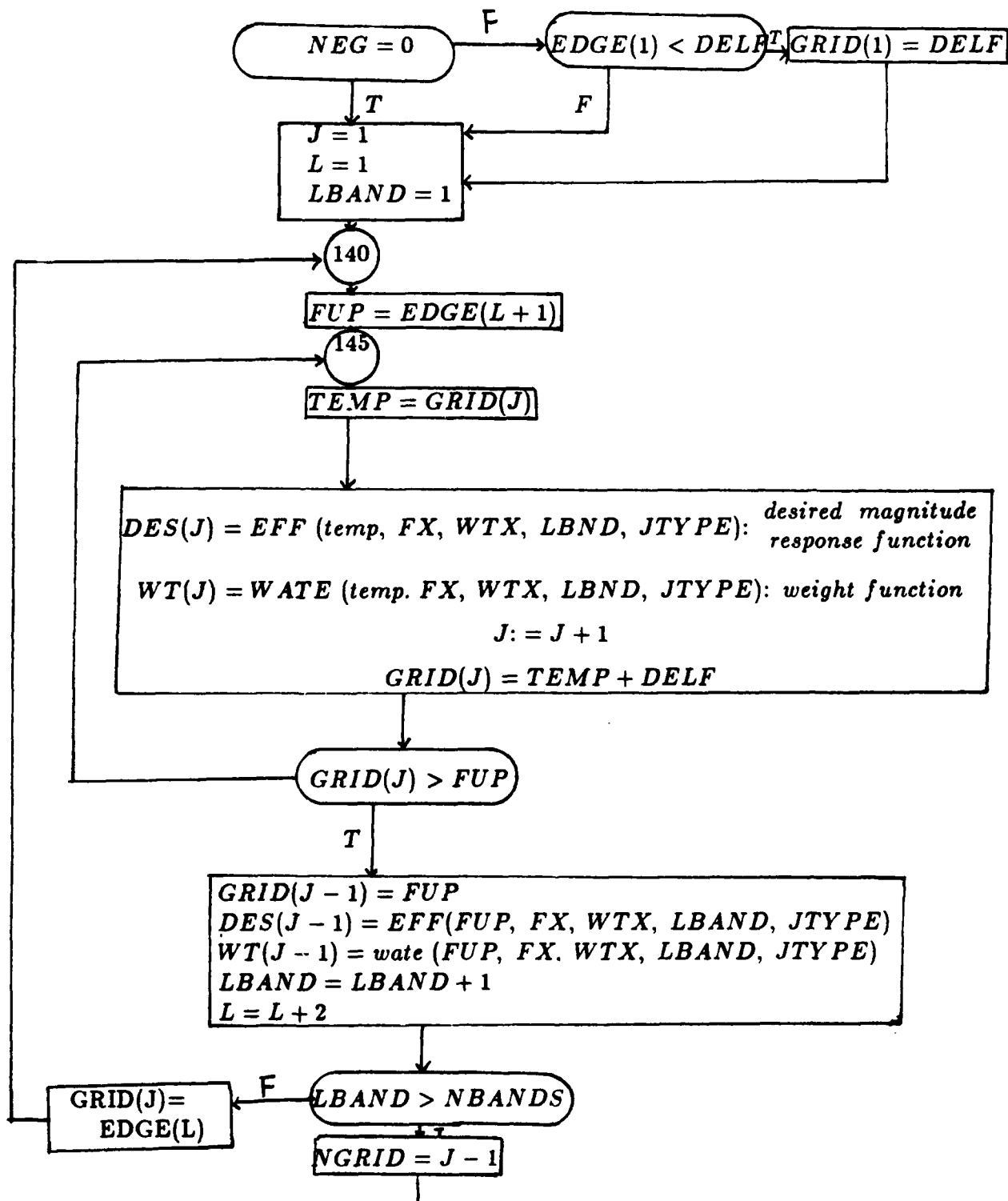


Figure 4: Detailed flow chart of REMEZ 1, showing all statement in the program - continued

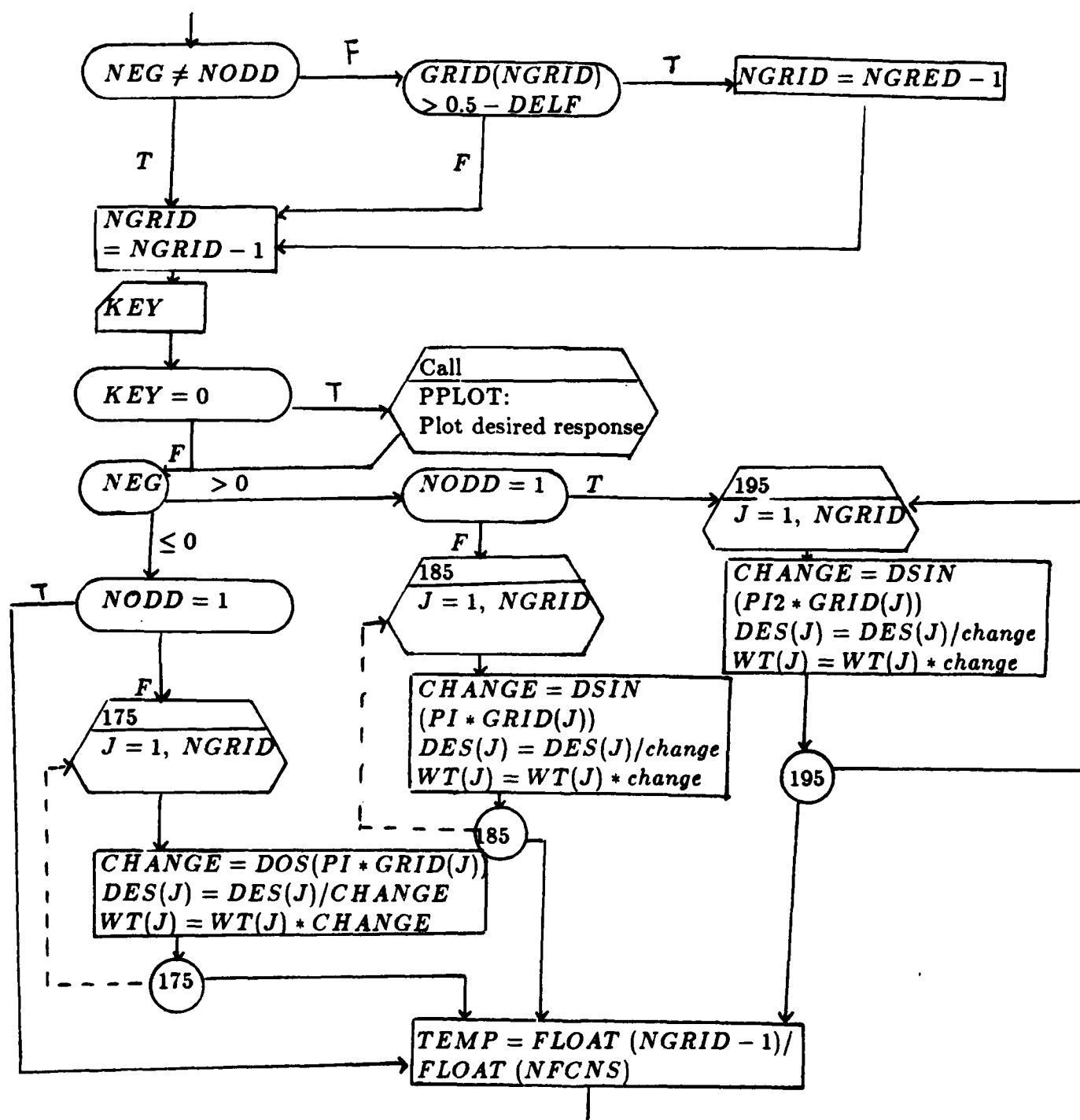
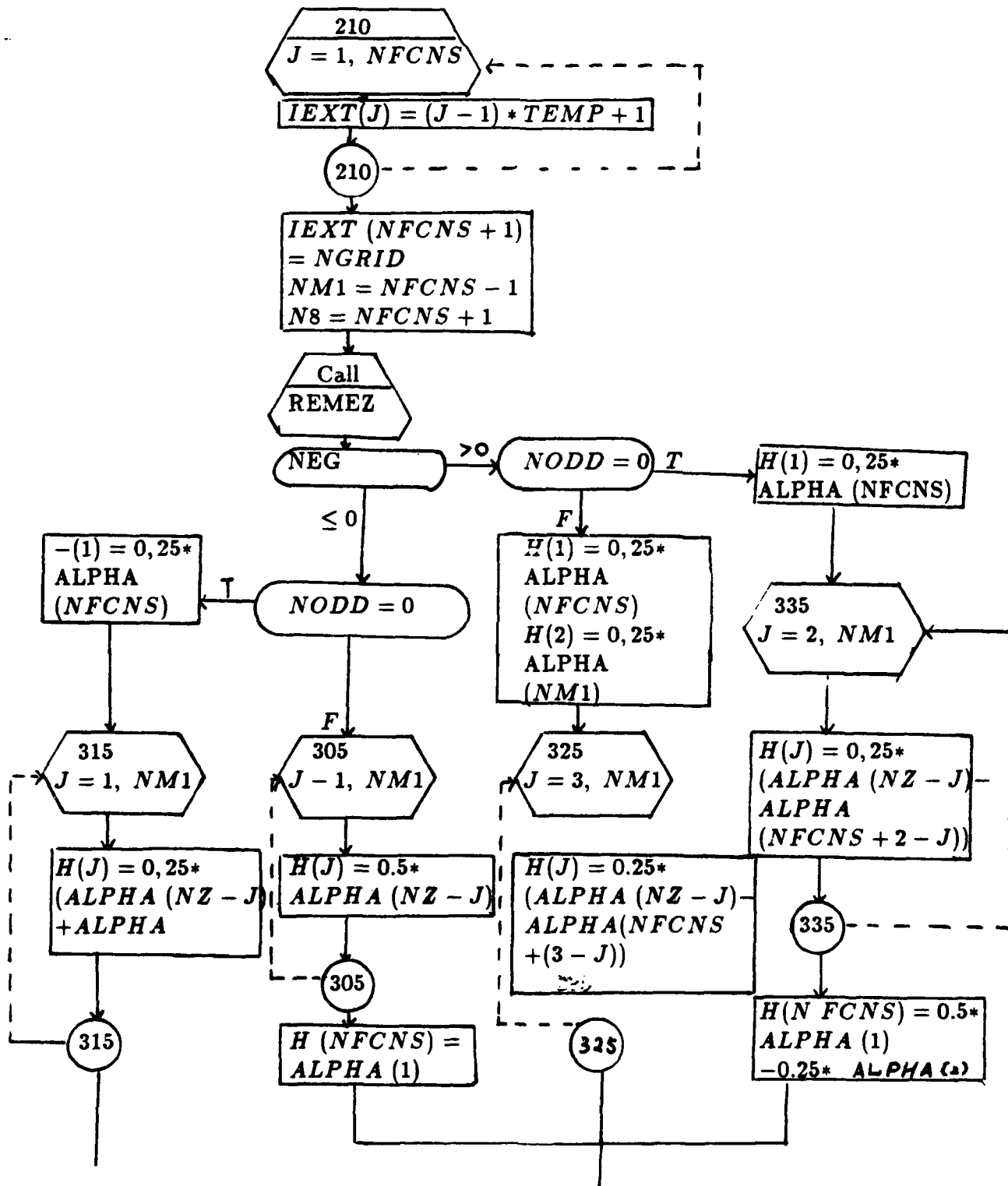


Figure 4: Detailed flow chart of REMEZ 1, showing all statements in the program - continued



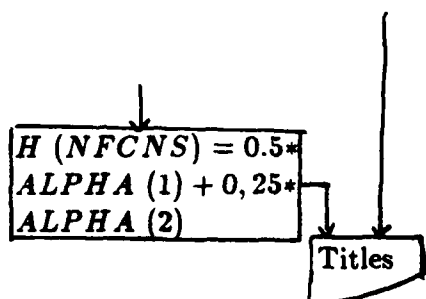
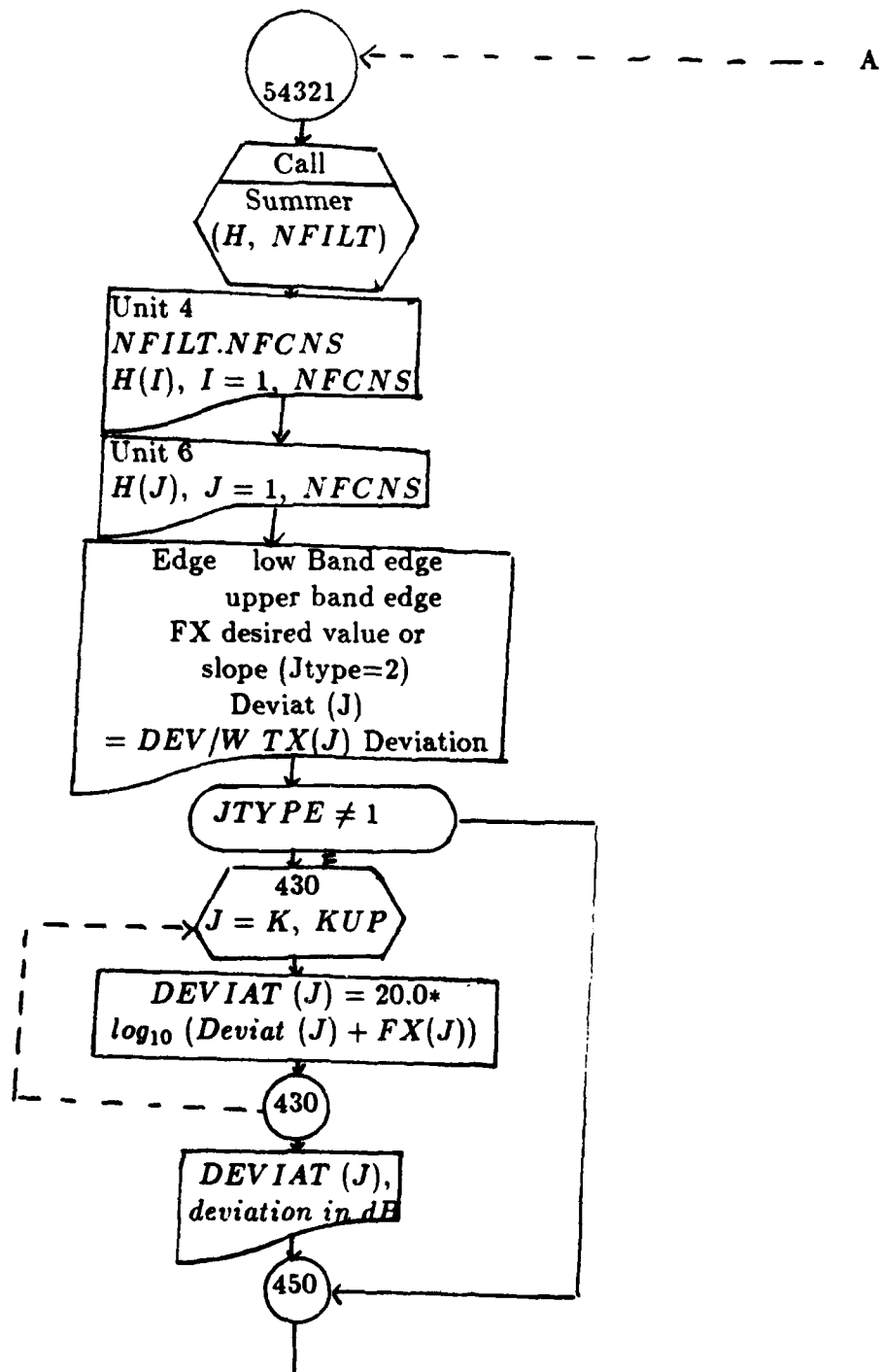


Figure 4: Detailed flow chart of REMEZ 1, showing all statements of the program - continued



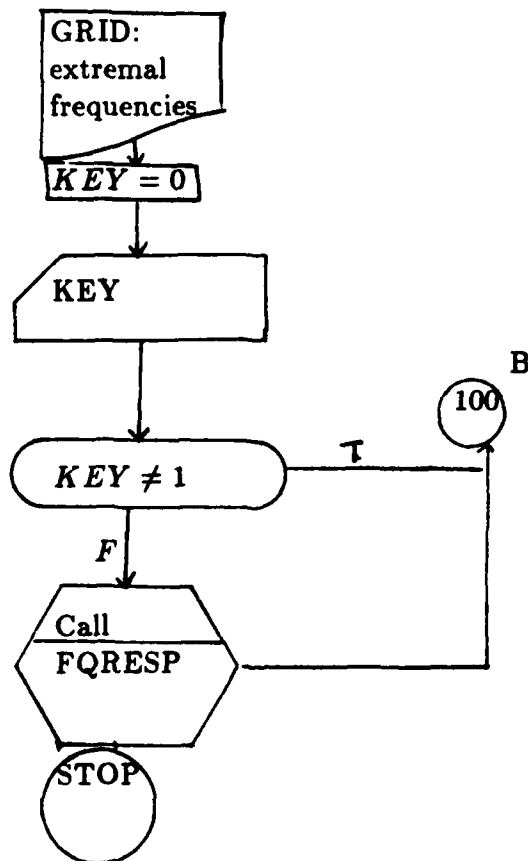
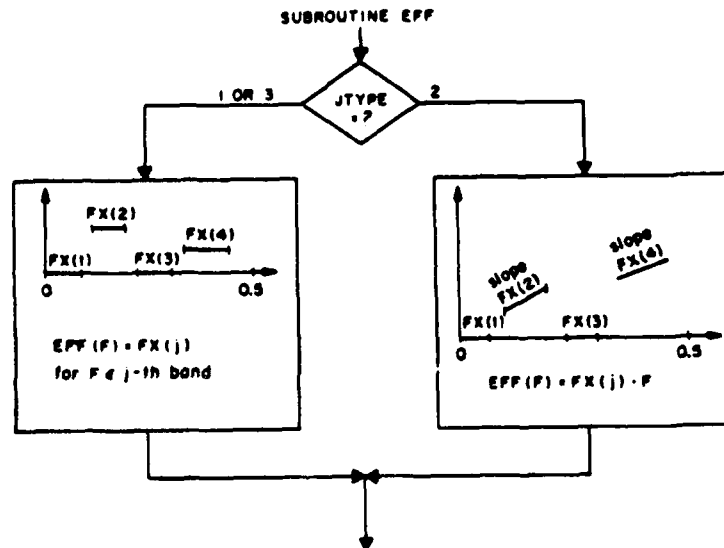


Figure 4: Detailed flow chart of REMEZ 1, showing all statements of the program - continued



Evaluates desired function at a grid point

Figure 5: Flow chart of Subroutine EFF



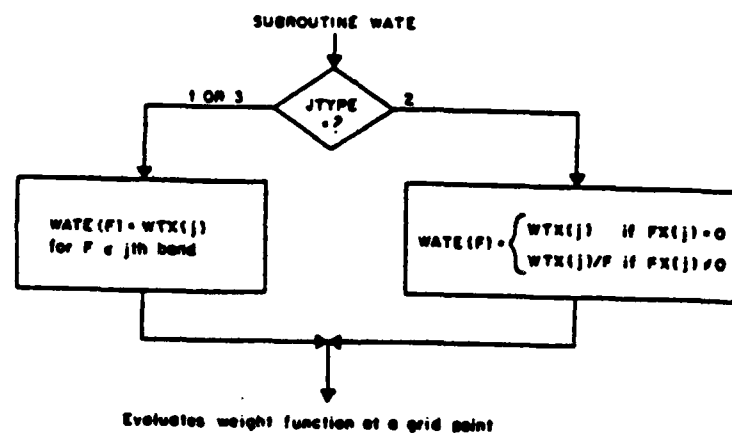


Figure 6: Flow chart of Subroutine WATE

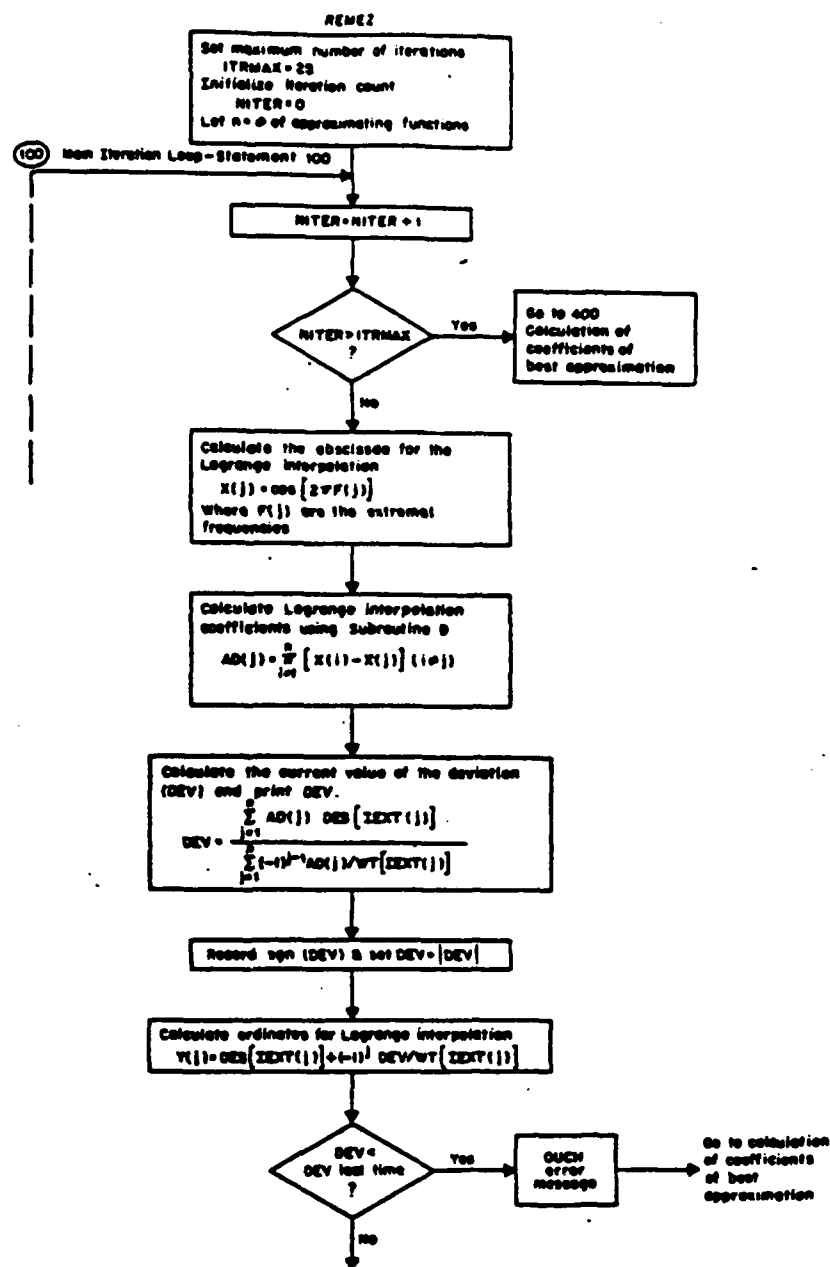


Figure 7: Flow chart of REMEZ



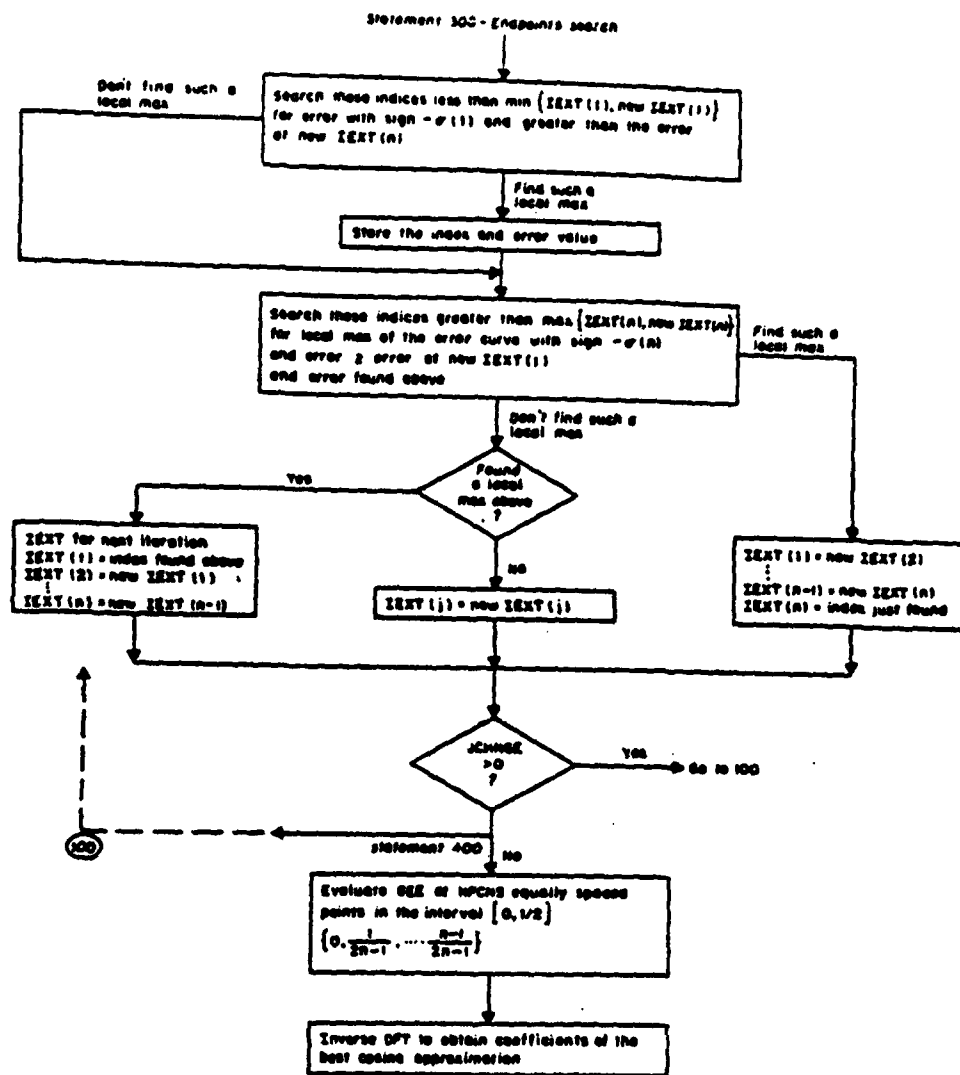


Figure 7: Flow chart of REMEZ - continued

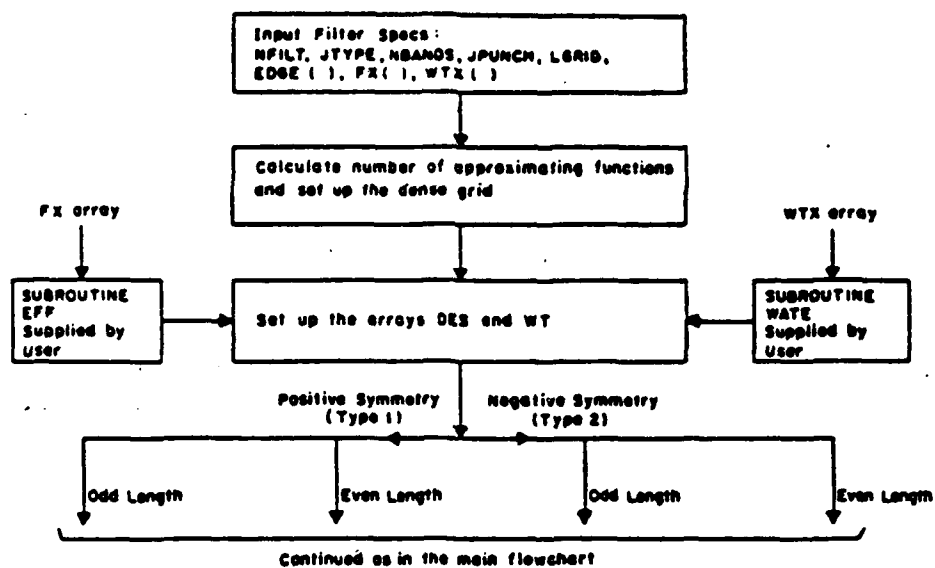


Figure 8: Flow chart for arbitrary magnitude filter design algorithm

### 3. Examples

The following figures show examples of REMEZ 1 outputs:

Figure 9: Low-pass filter with  $N = 24$ .

Figure 10: Bandpass filter with  $N = 32$ .

Figure 11: Bandpass filter with unequal weighting in the stopbands and  $N = 50$ .

Figure 12: Bandstop filter with  $N = 31$ .

Figure 13: Multiband filter with  $N = 55$ .

Figure 14: Differentiator with  $N = 32$ .

Figure 15: Hilbert transformer with  $N = 20$ .

Figure 16: Bandpass filter with arbitrary weighting characteristics.

```

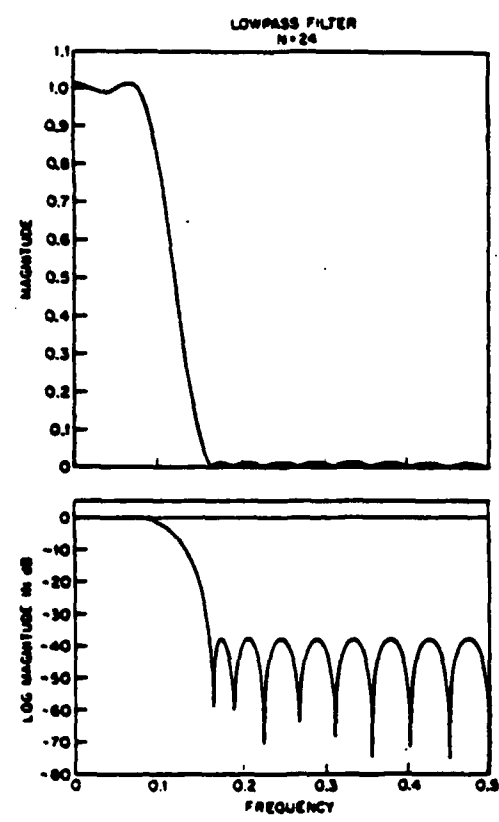
.....
                FIR IMPULSE RESPONSE (FIR)
                LINEAR PHASE DIGITAL FILTER DESIGN
                KAMAZ EXCHANGE ALGORITHM
                BANDPASS FILTER
                FILTER LENGTH = 24
                ***** IMPULSE RESPONSE *****
                NI 1) = 0.13740947E-02 = NI 24)
                NI 2) = 0.14340299E-01 = NI 23)
                NI 3) = 0.14364360E-01 = NI 22)
                NI 4) = 0.25419467E-02 = NI 21)
                NI 5) = -0.15723392E-01 = NI 20)
                NI 6) = -0.34065363E-01 = NI 19)
                NI 7) = -0.38112177E-01 = NI 18)
                NI 8) = -0.14025109E-01 = NI 17)
                NI 9) = 3.40309901E-01 = NI 16)
                NI 10) = 3.11948715E 00 = NI 15)
                NI 11) = 0.13050752E 00 = NI 14)
                NI 12) = 0.23394000E 00 = NI 13)

                BAND 1          BAND 2          BAND 3
                LOWER BAND EDGE 0.          0.10000000 0.4000
                UPPER BAND EDGE 0.00000000 0.50000000
                DESIRED VALUE    1.00000000 0.
                WEIGHTING        1.00000000 1.00000000
                DEVIATION        0.01243340 0.01243340
                DEVIATION IN DB -33.10403613 -33.10403613

                EXTREMAL FREQUENCIES
                0.          0.00000000 0.0077063 0.0000000 0.1600000
                0.1730200 0.4000750 0.4000375 0.4070000 0.5000000
                0.3787500 0.4256251 0.4751063

                .....
                TIME= 0.7694063 SECONDS
    
```

Output listing for an N = 24 low-pass filter.



Magnitude responses, on linear and log scales, for an N = 24 low-pass filter.

Figure 9: Low-pass filter with N=24

```

.....
FINITE IMPULSE RESPONSE (FIR)
LINEAR PHASE DIGITAL FILTER DESIGN
Remez Exchange Algorithm
BANDPASS FILTER
FILTER LENGTH = 32
***** IMPULSE RESPONSE *****
M( 1) = -0.5774641E-04 = M( 32)
M( 2) =  0.3402719E-03 = M( 31)
M( 3) =  0.7974354E-04 = M( 30)
M( 4) = -0.6514119E-02 = M( 29)
M( 5) =  0.1390852E-01 = M( 28)
M( 6) =  0.2235146E-02 = M( 27)
M( 7) = -0.1999406E-01 = M( 26)
M( 8) =  0.7136950E-02 = M( 25)
M( 9) = -0.3965736E-01 = M( 24)
M(10) =  0.1126011E-01 = M( 23)
M(11) =  0.6623264E-01 = M( 22)
M(12) = -0.1049722E-01 = M( 21)
M(13) =  0.4513613E-01 = M( 20)
M(14) = -0.1202439E  00 = M( 19)
M(15) = -0.2967857E  00 = M( 18)
M(16) =  0.3641697E  00 = M( 17)

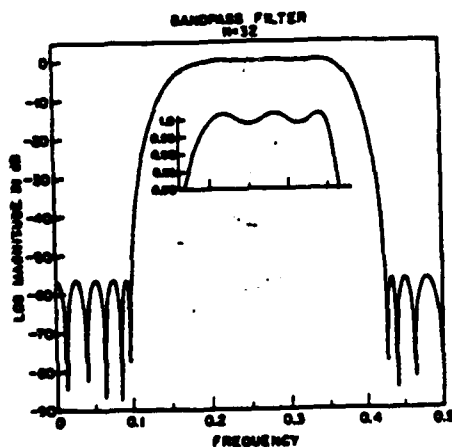
BAND 1          BAND 2          BAND 3          BAND
LOWER BAND EDGE 0.          0.20000000  0.42500000
UPPER BAND EDGE 0.10000000  0.35000000  0.50000000
DESIRED VALUE    0.          1.00000000  0.
WEIGHTING        10.00000000  1.00000000  10.00000000
DEVIATION         0.00151312  0.01513110  0.00151312
DEVIATION IN DB  -56.40294641 -36.40294641 -56.40294641

EXTERNAL FREQUENCIES
0.          0.0273437  0.0927344  0.0761719  0.0937500
0.10000000  0.20000000  0.4195312  0.4527344  0.2039644
0.3132012  0.3306719  0.35000000  0.4250000  0.4320125
0.4503900  0.4790675

*****
TIME= 0.0249025 SECONDS

```

Output listing for an N = 32 bandpass filter.



Log magnitude response for an N = 32 bandpass filter.

Figure 10: Bandpass filter with N=32



```

.....
FINITE IMPULSE RESPONSE (FIR)
LINEAR PHASE DIGITAL FILTER DESIGN
RAMEZ EXCHANGE ALGORITHM
BANDPASS FILTER
Filter Length = 51
***** IMPULSE RESPONSE *****
NI 1) = 0.00000000e+00 = NI 51)
NI 2) = 0.00000000e+00 = NI 50)
NI 3) = -0.31745250e-02 = NI 49)
NI 4) = -0.61100110e-02 = NI 48)
NI 5) = 0.74150645e-02 = NI 47)
NI 6) = 0.34364361e-02 = NI 46)
NI 7) = -0.11103735e-01 = NI 45)
NI 8) = -0.10101127e-01 = NI 44)
NI 9) = 0.69349206e-02 = NI 43)
NI 10) = 0.23400140e-02 = NI 42)
NI 11) = 0.26032332e-02 = NI 41)
NI 12) = 0.12421950e-01 = NI 40)
NI 13) = -0.20657142e-01 = NI 39)
NI 14) = -0.27169307e-01 = NI 38)
NI 15) = 0.32137130e-01 = NI 37)
NI 16) = 0.26185613e-01 = NI 36)
NI 17) = -0.20922861e-01 = NI 35)
NI 18) = -0.16761153e-02 = NI 34)
NI 19) = -0.22623357e-01 = NI 33)
NI 20) = -0.53326217e-01 = NI 32)
NI 21) = 0.30472539e-01 = NI 31)
NI 22) = 0.12315772e-01 = NI 30)
NI 23) = -0.15619221e-01 = NI 29)
NI 24) = -0.17713444e-01 = NI 28)
NI 25) = 0.13070169e-01 = NI 27)

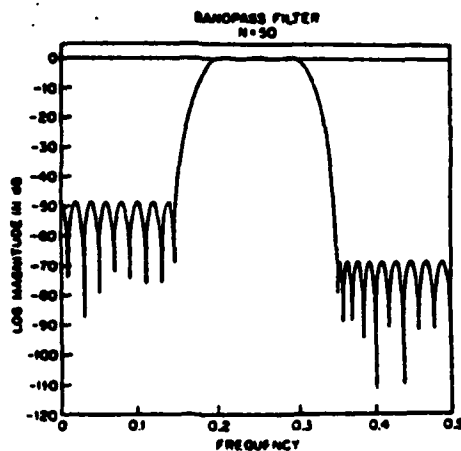
BAND 1          BAND 2          BAND 3          BAND
LOWER BAND EDGE 0.15080000 0.20030000 0.35000000
UPPER BAND EDGE 0.15080000 0.30030000 0.50030000
DESIRED VALUE    0.          1.00030000 3.
WEIGHTING        10.00000000 1.00000000 100.00000000
DEVIATION        0.00370509 3.03705040 3.00037050
DEVIATION IN DB -46.62412214 -26.62412214 -66.62412262

EXTREMAL FREQUENCIES
0.          0.00000000 0.00000000 0.06125000 0.00125000
0.16125000 0.12250000 0.14125000 0.15000000 0.20000000
0.21000000 0.22075000 0.25000000 0.27125000 0.29000000
0.30000000 0.35000000 0.35375000 0.36375000 0.37625000
0.39430000 0.40000000 0.42675000 0.44675000 0.46675000
0.48999999

*****
TIME= 2.9562969 SECONDS

```

Output listing for an  $N = 50$  bandpass filter with unequal weighting in the stopbands.



Log magnitude response of an  $N = 50$  bandpass filter with unequal weighting in the stopbands.

Figure 11: Bandpass filter with unequal weighting in the stopbands and  $N=50$

```

.....
FINITE IMPULSE RESPONSE (FIR)
LINEAR PHASE DIGITAL FILTER DESIGN
RUMER LAURENCE ALGORITHM
BANDPASS FILTER
Filter Length = 31
===== IMPULSE RESPONSE =====
NI 1) = -0.00760000e-04 = NI 31)
NI 2) = 0.10295000e-01 = NI 30)
NI 3) = -0.56002001e-02 = NI 29)
NI 4) = 0.52160000e-01 = NI 28)
NI 5) = 0.31550000e-02 = NI 27)
NI 6) = 0.03401227e-01 = NI 26)
NI 7) = 0.11600220e-01 = NI 25)
NI 8) = -0.07015000e-01 = NI 24)
NI 9) = 0.00000000e-02 = NI 23)
NI 10) = -0.07500000e-01 = NI 22)
NI 11) = -0.10000000e-01 = NI 21)
NI 12) = 0.00000000e-01 = NI 20)
NI 13) = -0.00000000e-02 = NI 19)
NI 14) = 0.01100000e-01 = NI 18)
NI 15) = 0.00000000e-02 = NI 17)
NI 16) = 0.05200000e-01 = NI 16)

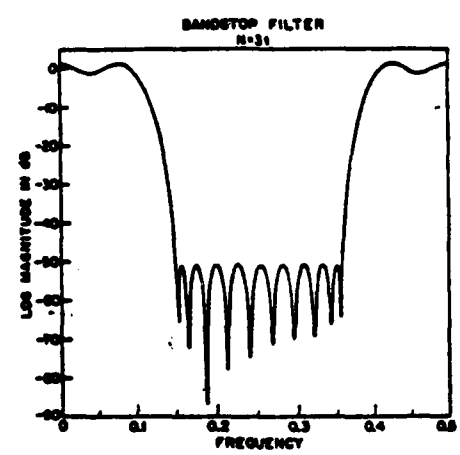
BAND 1 BAND 2 BAND 3 BAND
Lower Band Edge 0. 0.15000000 0.20000000
Upper Band Edge 0.10000000 0.30000000 0.50000000
Desired Value 0.00000000 0. 1.00000000
Weighting 0.00000000 0.00000000 1.00000000
Deviation 0.10000000 0.00000000 0.10000000
Deviation in dB -16.83153510 -50.01003562 -16.83153510

EXTERNAL FREQUENCIES
0. 0.00000000 0.0701250 0.1000000 0.1500000
0.10/0.125 0.0733000 0.2000000 0.2500000
0.20/0.125 0.3000000 0.3333333 0.3500000
0.40/0.000 0.0000000

=====
TIME = 1 1.6110156 S.C.MUS

```

Output listing for an N = 31 bandstop filter.



Log magnitude response for an N = 31 bandstop filter.

Figure 12: Bandstop filter with N=31

FINITE IMPULSE RESPONSE (FIR)  
 LINEAR PHASE DIGITAL FILTER DESIGN  
 KEMZ EXCHANGE ALGORITHM  
 BANDPASS FILTER  
 FILTER LENGTH = 55

\*\*\*\*\* IMPULSE RESPONSE \*\*\*\*\*

h( 1) =	0.00000000	h( 55) =
h( 2) =	0.00000000	h( 54) =
h( 3) =	0.00000000	h( 53) =
h( 4) =	0.00000000	h( 52) =
h( 5) =	0.00000000	h( 51) =
h( 6) =	0.00000000	h( 50) =
h( 7) =	0.00000000	h( 49) =
h( 8) =	0.00000000	h( 48) =
h( 9) =	0.00000000	h( 47) =
h(10) =	0.00000000	h( 46) =
h(11) =	0.00000000	h( 45) =
h(12) =	0.00000000	h( 44) =
h(13) =	0.00000000	h( 43) =
h(14) =	0.00000000	h( 42) =
h(15) =	0.00000000	h( 41) =
h(16) =	0.00000000	h( 40) =
h(17) =	0.00000000	h( 39) =
h(18) =	0.00000000	h( 38) =
h(19) =	0.00000000	h( 37) =
h(20) =	0.00000000	h( 36) =
h(21) =	0.00000000	h( 35) =
h(22) =	0.00000000	h( 34) =
h(23) =	0.00000000	h( 33) =
h(24) =	0.00000000	h( 32) =
h(25) =	0.00000000	h( 31) =
h(26) =	0.00000000	h( 30) =
h(27) =	0.00000000	h( 29) =
h(28) =	0.00000000	h( 28) =

	BAND 1	BAND 2	BAND 3	WALO
LOWER BAND EDGE	0.00000000	0.00000000	0.00000000	0.00000000
UPPER BAND EDGE	0.00000000	0.00000000	0.00000000	0.00000000
DESIRED VALUE	0.00000000	0.00000000	0.00000000	0.00000000
WEIGHTING	0.00000000	0.00000000	0.00000000	0.00000000
DEVIATION	0.00000000	0.00000000	0.00000000	0.00000000
DEVIATION IN DB	-40.00000000	-40.00000000	-40.00000000	-40.00000000

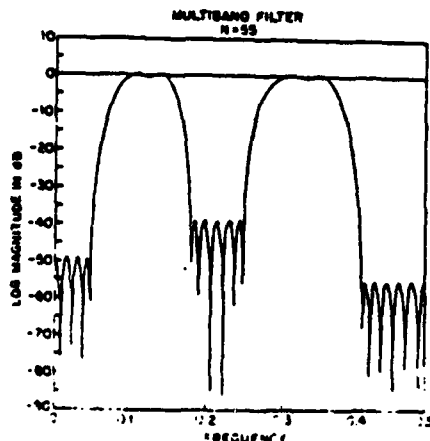
	BAND 3	BAND 0
LOWER BAND EDGE	0.00000000	0.00000000
UPPER BAND EDGE	0.00000000	0.00000000
DESIRED VALUE	0.00000000	0.00000000
WEIGHTING	0.00000000	0.00000000
DEVIATION	0.00000000	0.00000000
DEVIATION IN DB	-40.00000000	-40.00000000

EXTREMAL FREQUENCIES

0.	0.0167+11	0.0323+01	0.0444+29	0.0500+00
0.1000000	0.1000000	0.1000000	0.1000000	0.1000000
0.1000000	0.1000000	0.1000000	0.1000000	0.1000000
0.2000000	0.2000000	0.2000000	0.2000000	0.2000000
0.3000000	0.3000000	0.3000000	0.3000000	0.3000000
0.4000000	0.4000000	0.4000000	0.4000000	0.4000000

TIME= 3.8164219 SECONDS

Output listing for an N = 55 multiband filter.



Log magnitude response for an N = 55 multiband filter.

Figure 13: Multiband filter with N=55

```

.....
FINITE IMPULSE RESPONSE (FIR)
LINEAR PHASE DIGITAL FILTER DESIGN
REMEZ EXCHANGE ALGORITHM
DIFFERENTIATION
FILTER LENGTH = 32
***** IMPULSE RESPONSE *****
M( 1) = -0.00713091E-03 = -M( 32)
M( 2) = 0.05633643E-03 = -M( 31)
M( 3) = -0.42618549E-03 = -M( 30)
M( 4) = 0.39981518E-03 = -M( 29)
M( 5) = -0.44637273E-03 = -M( 28)
M( 6) = 0.44369658E-03 = -M( 27)
M( 7) = -0.54636961E-03 = -M( 26)
M( 8) = 0.73277031E-03 = -M( 25)
M( 9) = -0.93882661E-03 = -M( 24)
M( 10) = 0.12278842E-02 = -M( 23)
M( 11) = -0.17012620E-02 = -M( 22)
M( 12) = 0.25272341E-02 = -M( 21)
M( 13) = -0.41001159E-02 = -M( 20)
M( 14) = 0.61294355E-02 = -M( 19)
M( 15) = -0.22539897E-01 = -M( 18)
M( 16) = 0.28266539E 00 = -M( 17)

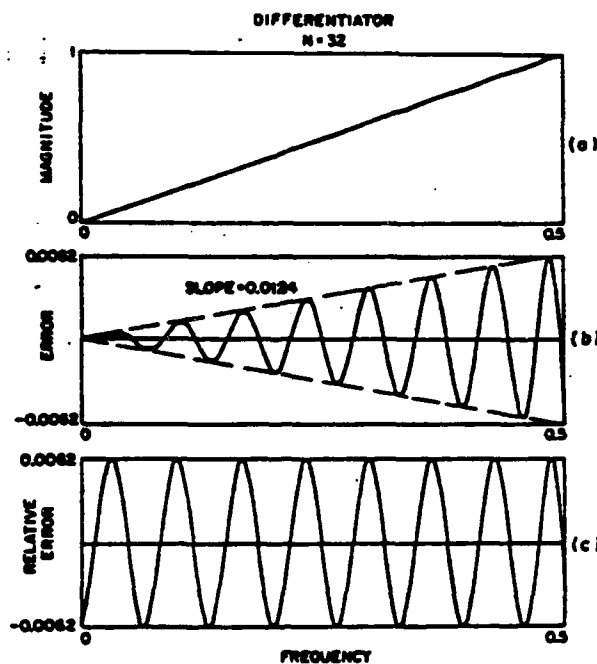
      BAND 1      BAND
LOWER BAND EDGE      0.
UPPER BAND EDGE      0.50000000
DESIRED SLOPE        1.00000000
WEIGHTING             1.00000000
DEVIATION             0.00000000

EXTREMAL FREQUENCIES
0.0014531  0.0332031  0.0864062  0.0956894  0.1328125
0.1000000  0.1972050  0.2300000  0.2636719  0.2966750
0.3330761  0.3866014  0.3900000  0.4277300  0.4500000
0.4863281  0.5000000

*****
TIME = 1.1072656 SECONDS

```

Output listing for an  $N = 32$  differentiator.



Magnitude and error responses for an  $N = 32$  differentiator.

Figure 14. Differentiator with  $N=32$

```

.....
FINITE IMPULSE RESPONSE (FIR)
LINEAR PHASE DIGITAL FILTER DESIGN
Remez algorithm
HILBERT TRANSFORMER
FILTER LENGTH = 20
***** IMPULSE RESPONSE *****
HI( 1) = 0.00000000e+00 = -HI( 20)
HI( 2) = 0.10173287e-01 = -HI( 19)
HI( 3) = 0.204692437e-01 = -HI( 18)
HI( 4) = 0.23736342e-01 = -HI( 17)
HI( 5) = 0.30492300e-01 = -HI( 16)
HI( 6) = 0.35333100e-01 = -HI( 15)
HI( 7) = 0.70542752e-01 = -HI( 14)
HI( 8) = 0.11923755e 00 = -HI( 13)
HI( 9) = 0.20664125e 00 = -HI( 12)
HI( 10) = 0.63675619e 00 = -HI( 11)

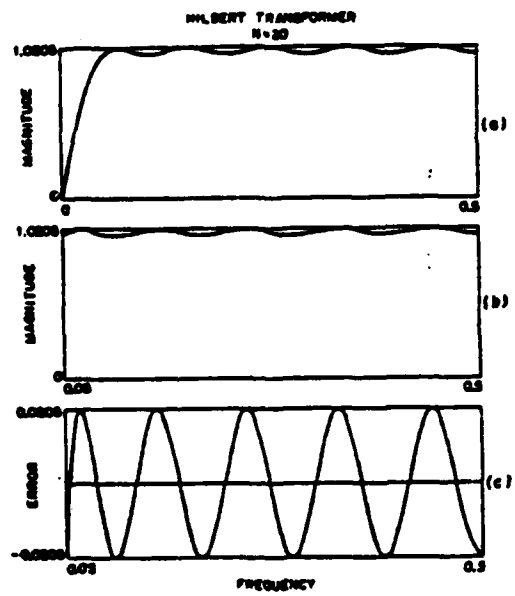
BAND L BAND
LOWER BAND EDGE 0.00000000
UPPER BAND EDGE 0.50000000
DESIREU VALUE 1.00000000
WEIGHTING 1.00000000
DEVIATION 0.00000000

EXTERNAL FREQUENCIES
0.0530000 0.0696250 0.1000000 0.1337500
0.2637500 0.2937500 0.3467500 0.3767500
0.4500000

1
.....
TIME= 0.4425137 SEC.10S

```

Output listing for an  $N = 20$  Hilbert transformer.



Magnitude and error responses for an  $N = 20$  Hilbert transformer.

Figure 15: Hilbert transformer with  $N=20$

.....

FINITE IMPULSE RESPONSE (FIR)  
 LINEAR PHASE DIGITAL FILTER DESIGN  
 RENEZ EXCHANGE ALGORITHM  
 BANDPASS FILTER

FILTER LENGTH = 128  
 \*\*\*\*\* IMPULSE RESPONSE \*\*\*\*\*

h( 1) =	-0.28862533E-02	=	h(128)
h( 2) =	0.60816867E-03	=	h(127)
h( 3) =	0.60820240E-03	=	h(126)
h( 4) =	0.67255829E-03	=	h(125)
h( 5) =	0.93766487E-03	=	h(124)
h( 6) =	0.68979566E-03	=	h(123)
h( 7) =	-0.15411817E-03	=	h(122)
h( 8) =	-0.97912618E-03	=	h(121)
h( 9) =	-0.18918514E-02	=	h(120)
h(10) =	-0.39942515E-03	=	h(119)
h(11) =	0.76848653E-03	=	h(118)
h(12) =	0.19240592E-02	=	h(117)
h(13) =	0.15246961E-02	=	h(116)
h(14) =	0.72599190E-03	=	h(115)
h(15) =	-0.40411612E-03	=	h(114)
h(16) =	-0.11471398E-02	=	h(113)
h(17) =	-0.10636777E-02	=	h(112)
h(18) =	-0.33459668E-03	=	h(111)
h(19) =	0.39046910E-03	=	h(110)
h(20) =	0.55175241E-03	=	h(109)
h(21) =	0.29497579E-03	=	h(108)
h(22) =	0.14940618E-04	=	h(107)
h(23) =	0.24607373E-03	=	h(106)
h(24) =	0.10619183E-02	=	h(105)
h(25) =	0.15913030E-02	=	h(104)
h(26) =	0.93133048E-03	=	h(103)
h(27) =	-0.11981893E-02	=	h(102)
h(28) =	-0.35016186E-02	=	h(101)
h(29) =	-0.44120789E-02	=	h(100)
h(30) =	-0.21495669E-02	=	h( 99)
h(31) =	0.26225995E-02	=	h( 98)
h(32) =	0.73584645E-02	=	h( 97)
h(33) =	0.06703702E-02	=	h( 96)
h(34) =	0.40213210E-02	=	h( 95)
h(35) =	-0.64717526E-02	=	h( 94)
h(36) =	-0.12166497E-01	=	h( 93)
h(37) =	-0.13546156E-01	=	h( 92)
h(38) =	-0.62183130E-02	=	h( 91)
h(39) =	0.67914337E-02	=	h( 90)
h(40) =	0.17967235E-01	=	h( 89)
h(41) =	0.19919913E-01	=	h( 88)
h(42) =	0.87717150E-02	=	h( 87)
h(43) =	-0.93673921E-02	=	h( 86)
h(44) =	-0.24259475E-01	=	h( 85)
h(45) =	-0.25988771E-01	=	h( 84)
h(46) =	-0.11429030E-01	=	h( 83)
h(47) =	0.12821631E-01	=	h( 82)
h(48) =	0.38664461E-01	=	h( 81)
h(49) =	0.32234460E-01	=	h( 80)
h(50) =	0.14018583E-01	=	h( 79)
h(51) =	-0.16528243E-01	=	h( 78)
h(52) =	-0.38548266E-01	=	h( 77)
h(53) =	-0.37984748E-01	=	h( 76)
h(54) =	-0.18241339E-01	=	h( 75)
h(55) =	0.16649269E-01	=	h( 74)
h(56) =	0.41344275E-01	=	h( 73)
h(57) =	0.42361176E-01	=	h( 72)
h(58) =	0.17929827E-01	=	h( 71)
h(59) =	-0.10140443E-01	=	h( 70)
h(60) =	-0.46548948E-01	=	h( 69)
h(61) =	-0.45118319E-01	=	h( 68)
h(62) =	-0.18956116E-01	=	h( 67)
h(63) =	0.18956116E-01	=	h( 66)
h(64) =	0.45849782E-01	=	h( 65)

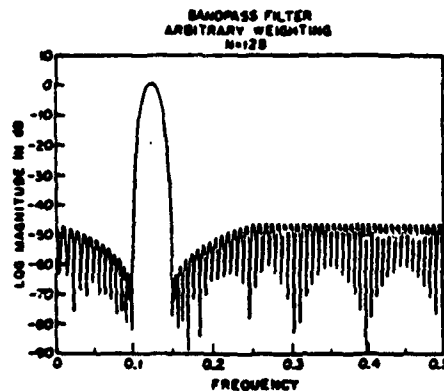
Output listing for an N = 128 bandpass filter with  
 arbitrary weighting characteristics.

Figure 16: Bandpass filter with arbitrary weighting characteristics

	PARAM 1	PARAM 2	PARAM 3	PARAM 4
LOWER BAND EDGE	0.	0.12770000	0.12770000	0.25000000
UPPER BAND EDGE	0.10000000	0.17000000	0.25000000	0.50000000
DESIGN VALUE	0.	1.00000000	0.	0.
WEIGHTING	10.00000000	1.00000000	10.00000000	10.00000000
DEVIATION	0.000001%	0.000001%	0.000001%	0.000001%
DEVIATION IN DB	-46.01027145	-25.01027145	-46.01027145	-46.01027145

EXTREMAL FREQUENCIES

0.	0.0102539	0.0200149	0.0280086	0.0371094
0.0444102	0.0532227	0.0610392	0.0686477	0.0761719
0.0834961	0.0898617	0.0992140	0.0996329	0.1000000
0.1200000	0.1248879	0.1300000	0.1500000	0.1514644
0.1544020	0.1602539	0.1644015	0.1736374	0.1807617
0.1800000	0.1940904	0.2027109	0.2119714	0.2193359
0.2271644	0.2349009	0.2427776	0.2509766	0.2592773
0.2670899	0.2749023	0.2827149	0.2910156	0.2988281
0.3066406	0.3144531	0.3222444	0.3309464	0.3393789
0.3461914	0.3540879	0.3614164	0.3696289	0.3779297
0.3857422	0.3935467	0.4013672	0.4091797	0.4169922
0.4252430	0.4331055	0.4409140	0.4487385	0.4565630
0.4644555	0.4721680	0.4804487	0.4887812	0.4968937



Log magnitude response for an  $N = 128$  bandpass filter with arbitrary weighting characteristics.

$$W(f) = \begin{cases} \frac{10}{1 - 9f} & 0 \leq f < 0.1 \\ 1 & 0.12 \leq f < 0.13 \\ \frac{10}{9f - 1.25} & 0.15 \leq f < 0.25 \\ 10 & 0.25 \leq f < 0.5. \end{cases}$$

Figure 16: Bandpass filter with arbitrary weighting characteristics (continued)

## **Description of Filtering/Spectral Analysis Program (IV)**

by  
Austin F.S. Lee, Ph.D.  
and  
Ralph B. D'Agostino, Ph.D.

The spectral analysis portion of the "filter and spectral analysis program" is presented here in a flow-chart form. The analysis used the maximum entropy technique of J.P. Bing and the computational program was comprised of the following routines: BPEC, NORM, PEAK, and SPEC, etc.



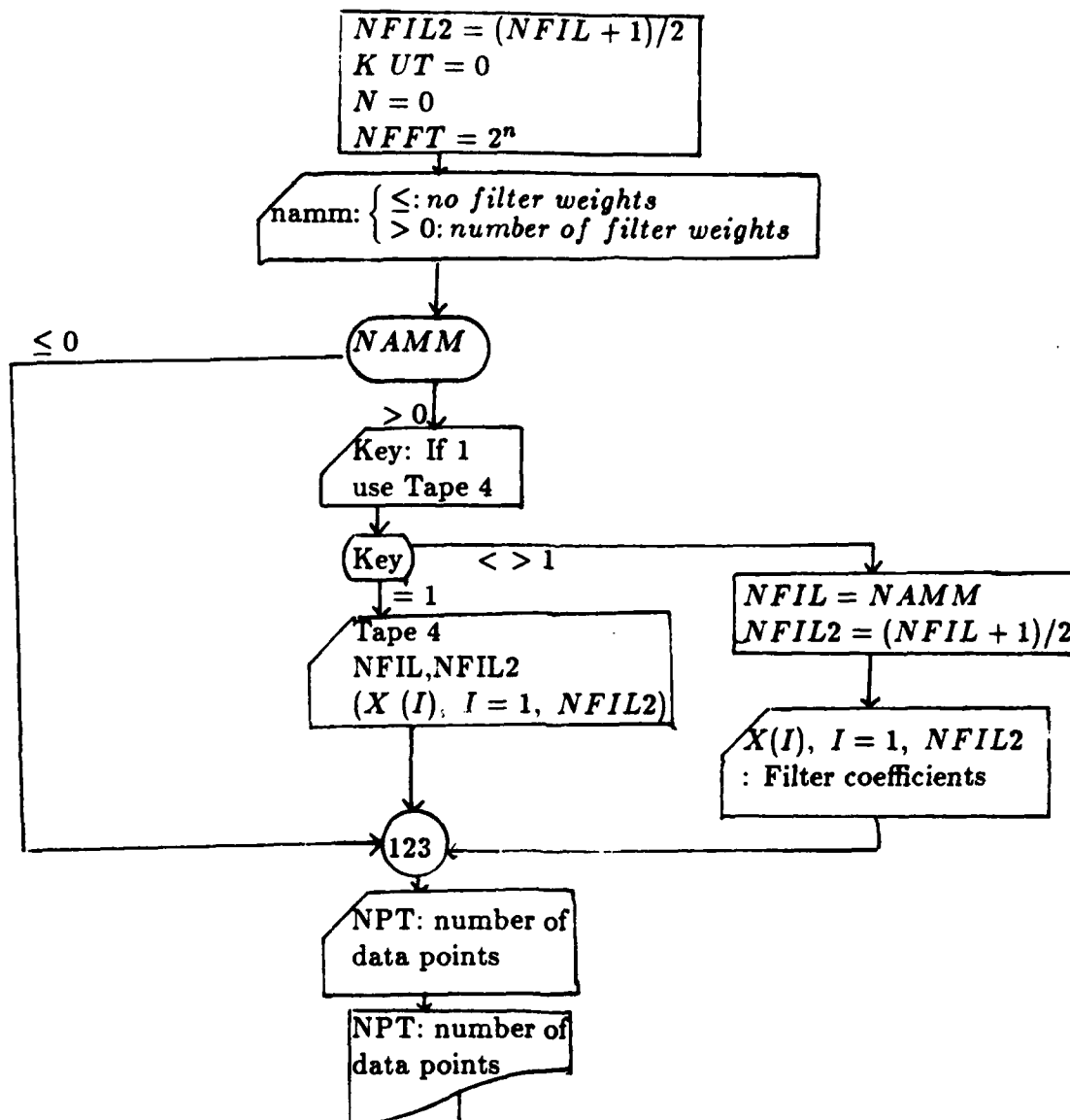


Figure 1: Detailed flow chart of FAST, showing all statements in the program.

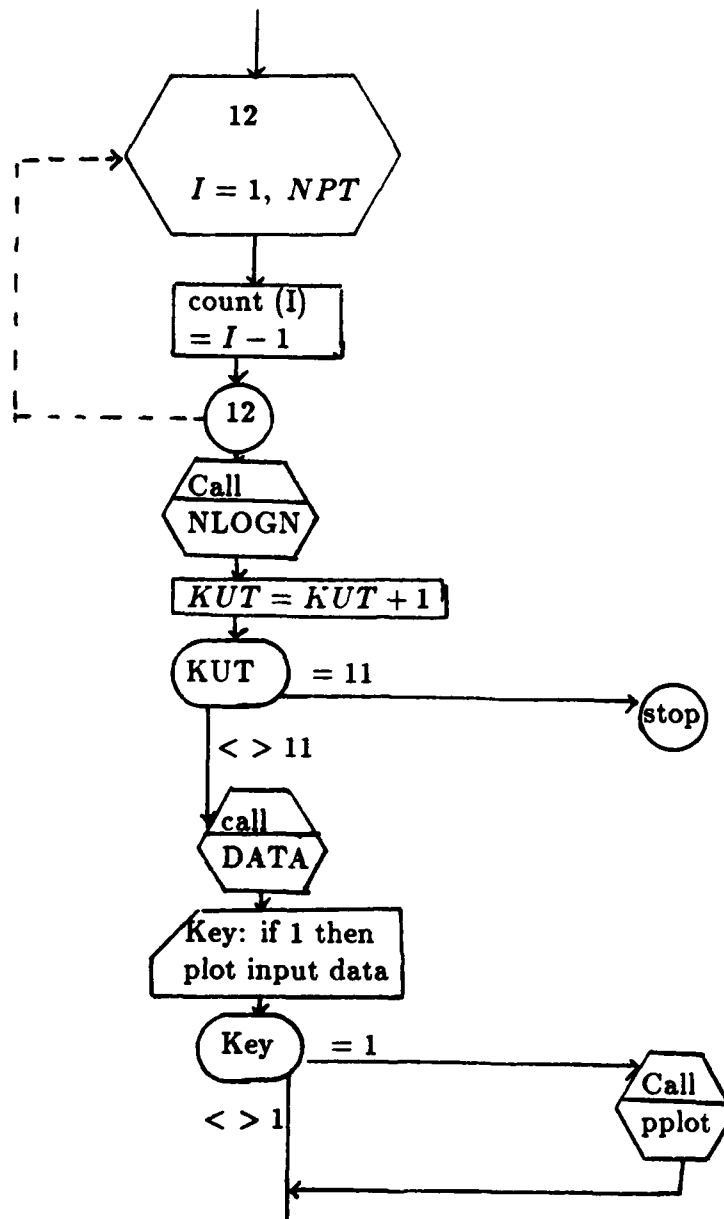


Figure 1: FAST continued

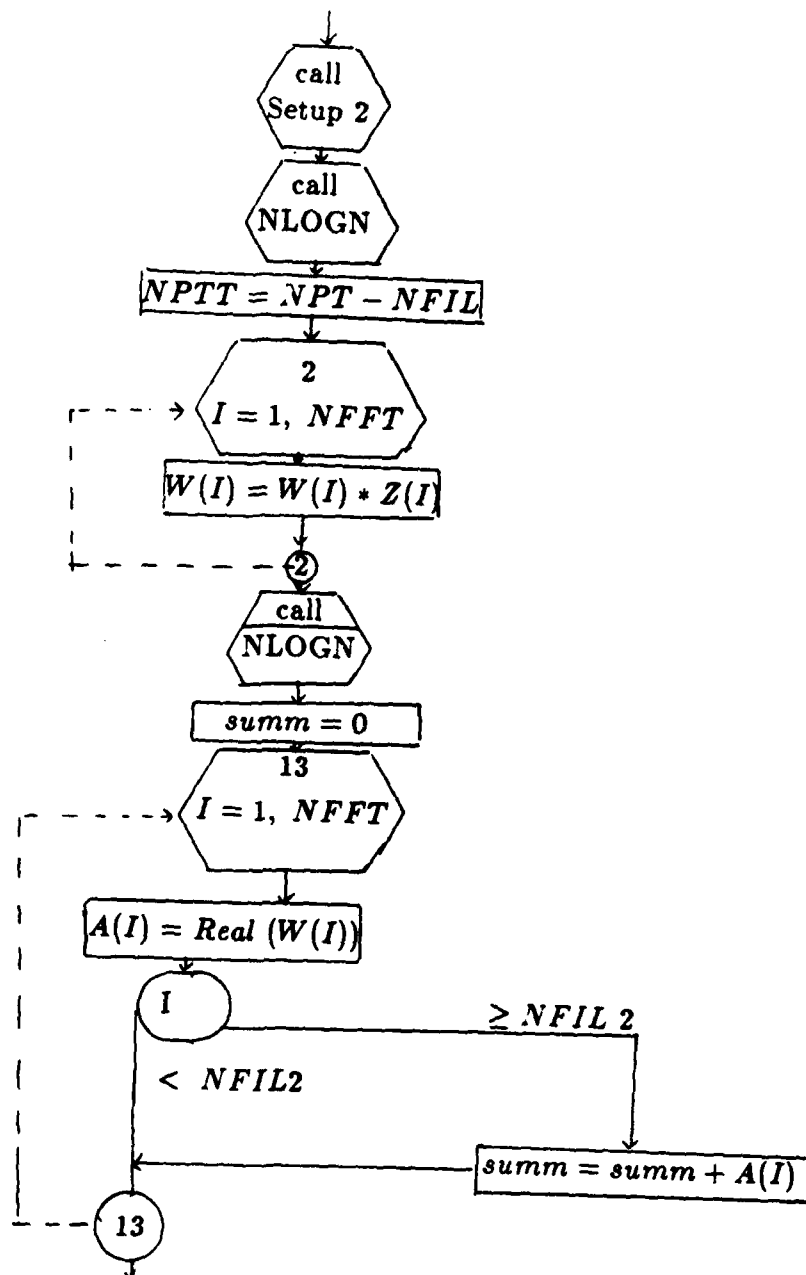


Figure 1: FAST continued

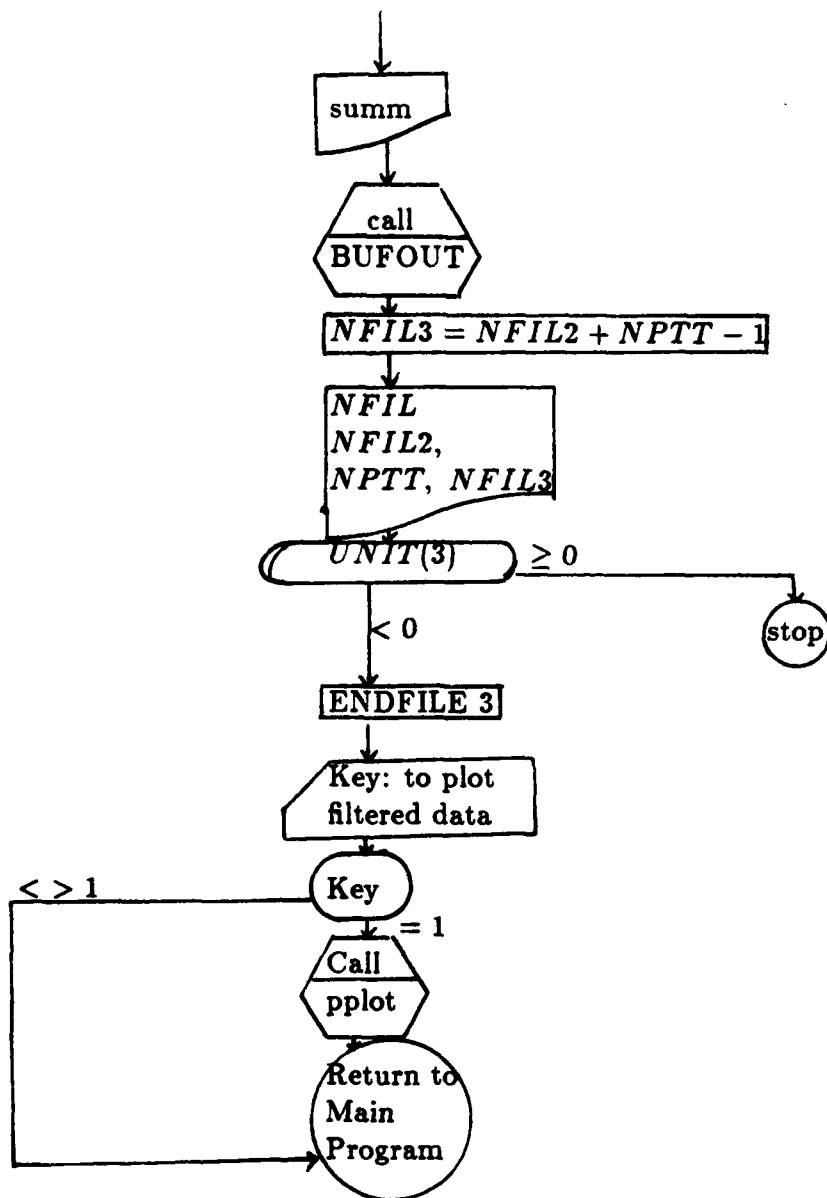


Figure 1: FAST continued

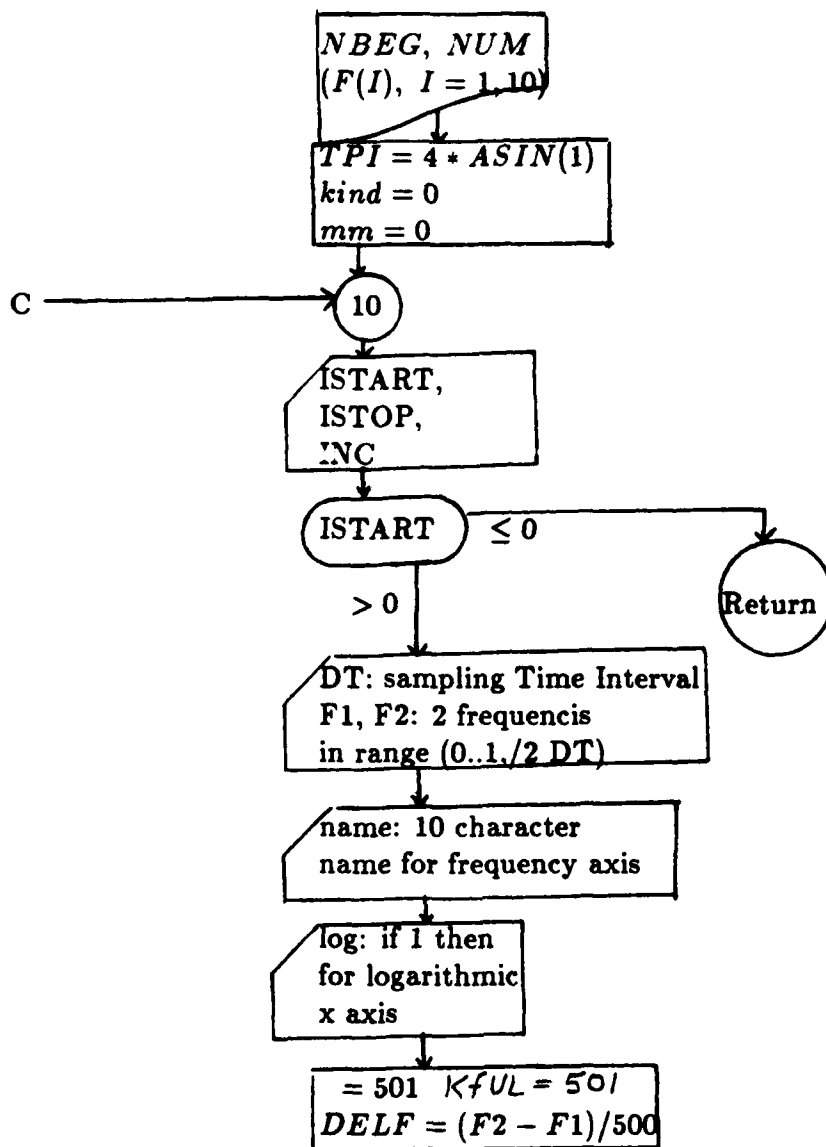


Figure 2: Detailed flow chart of SPECTRA showing all statements in the program.

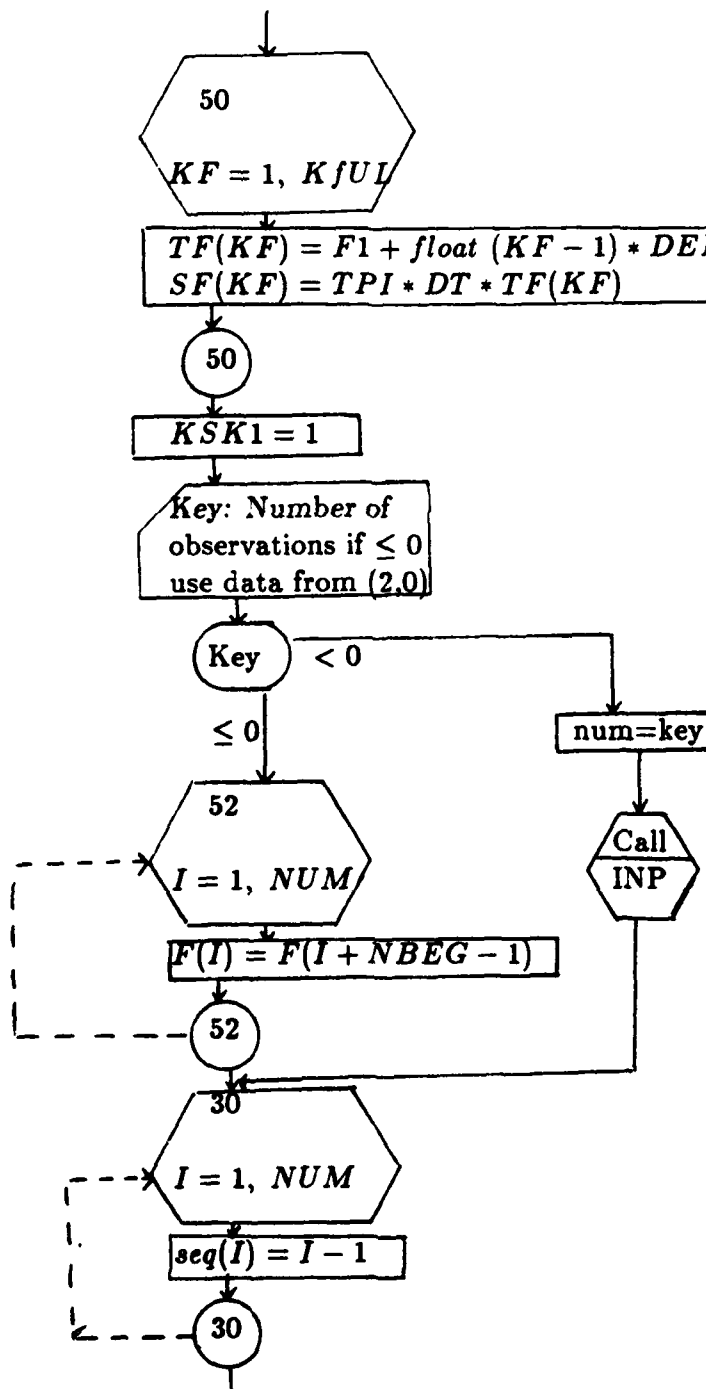


Figure 2: SPECTRA continued

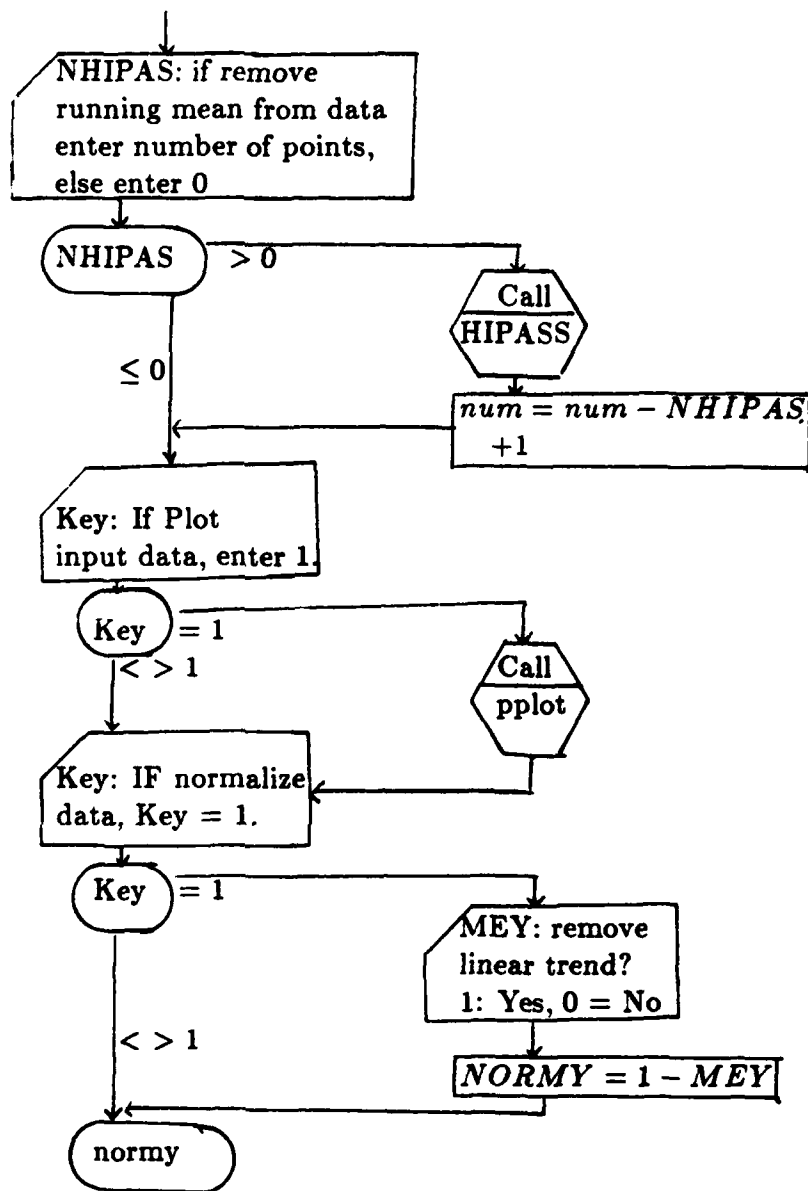


Figure 2: SPECTRA continued

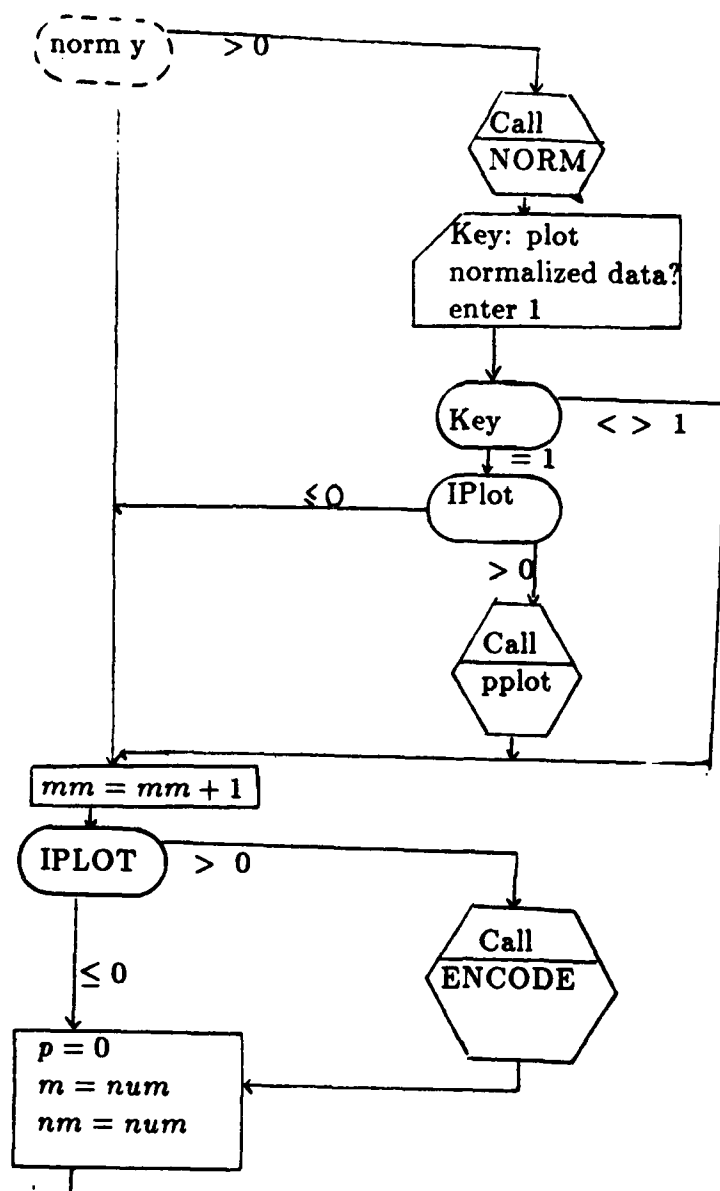


Figure 2: SPECTRA continued



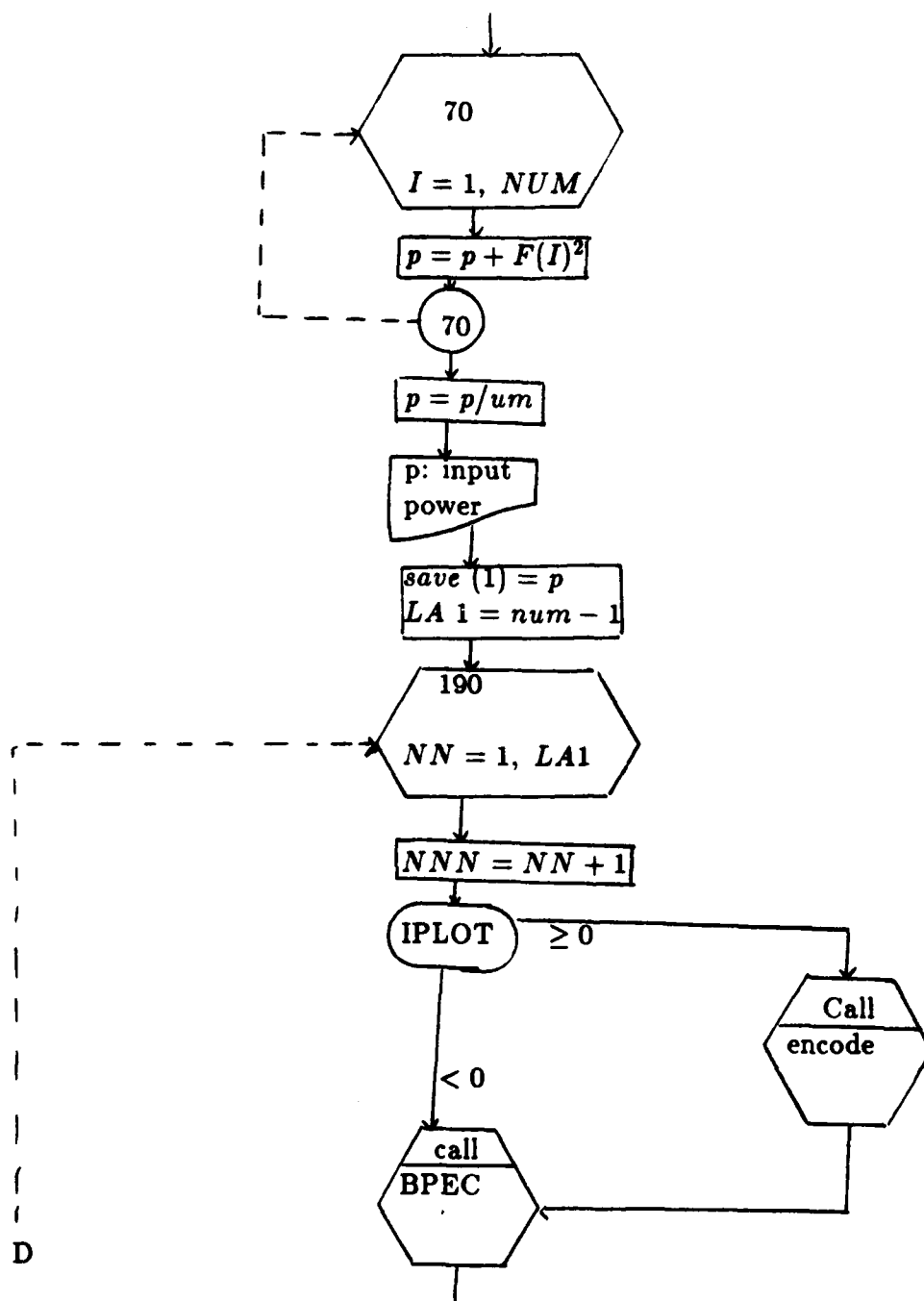


Figure 2: SPECTRA continued

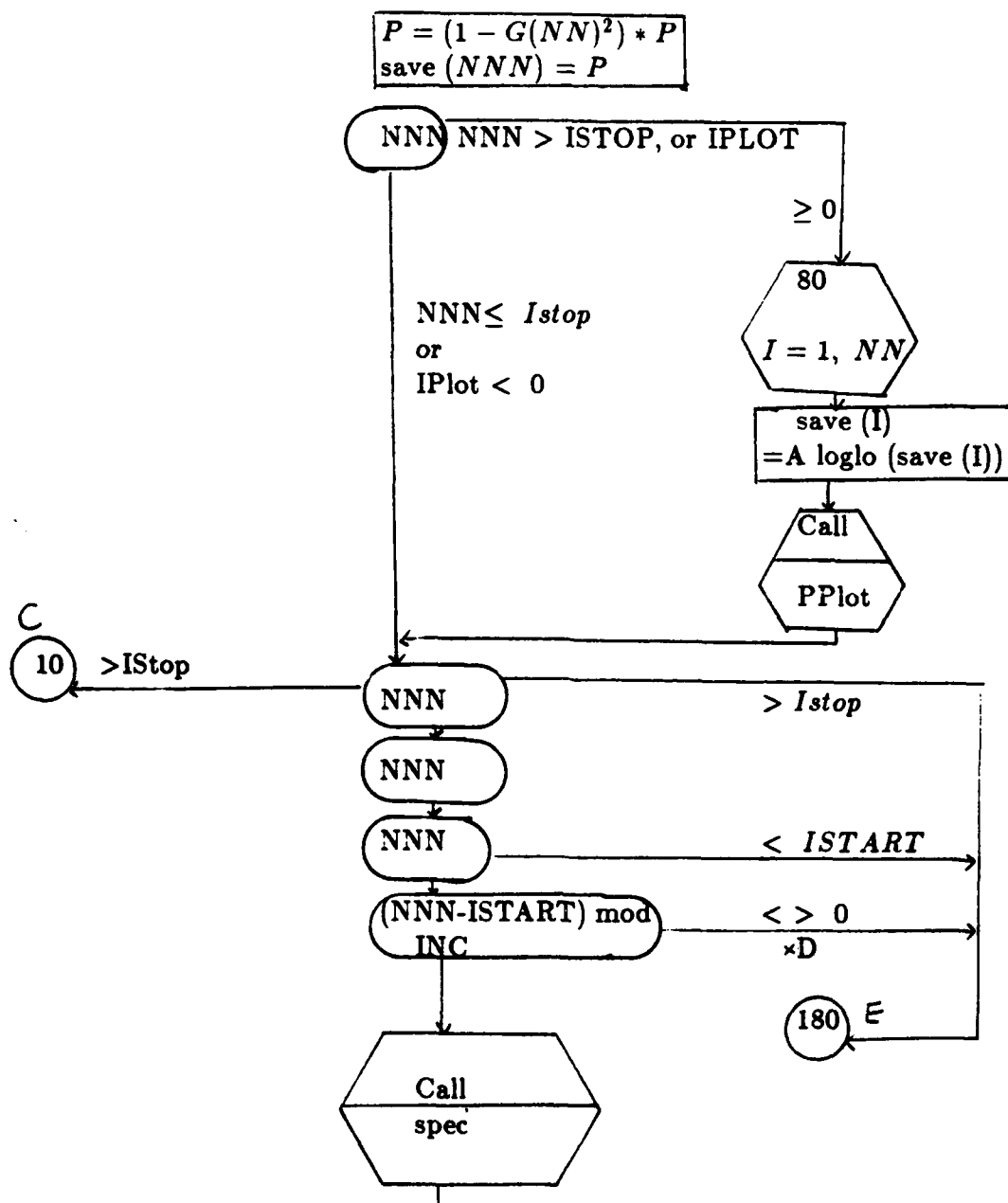


Figure 2: SPECTRA continued

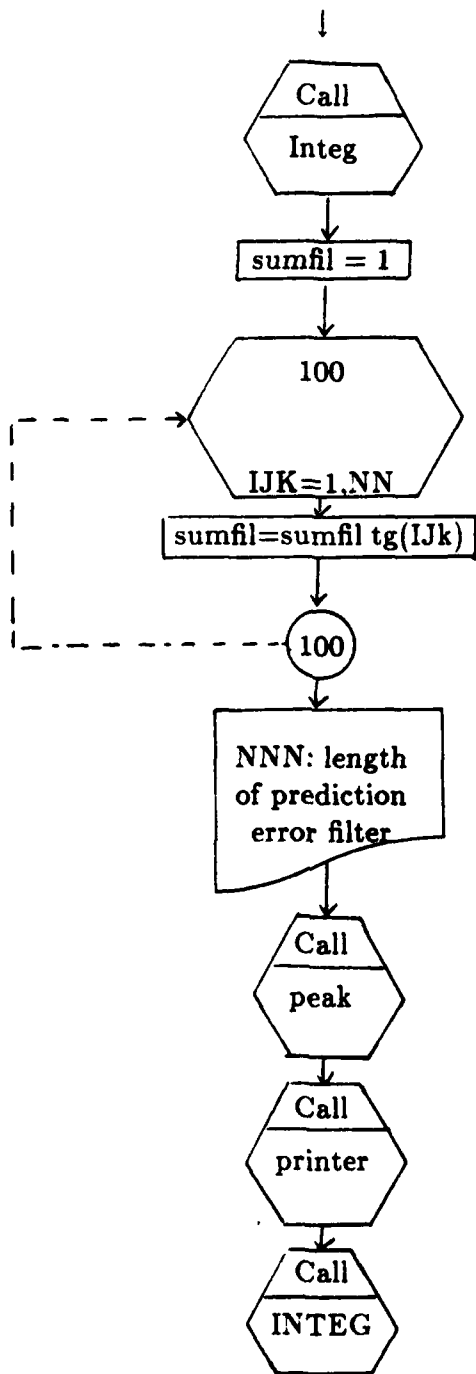


Figure 2 SPECTRA continued

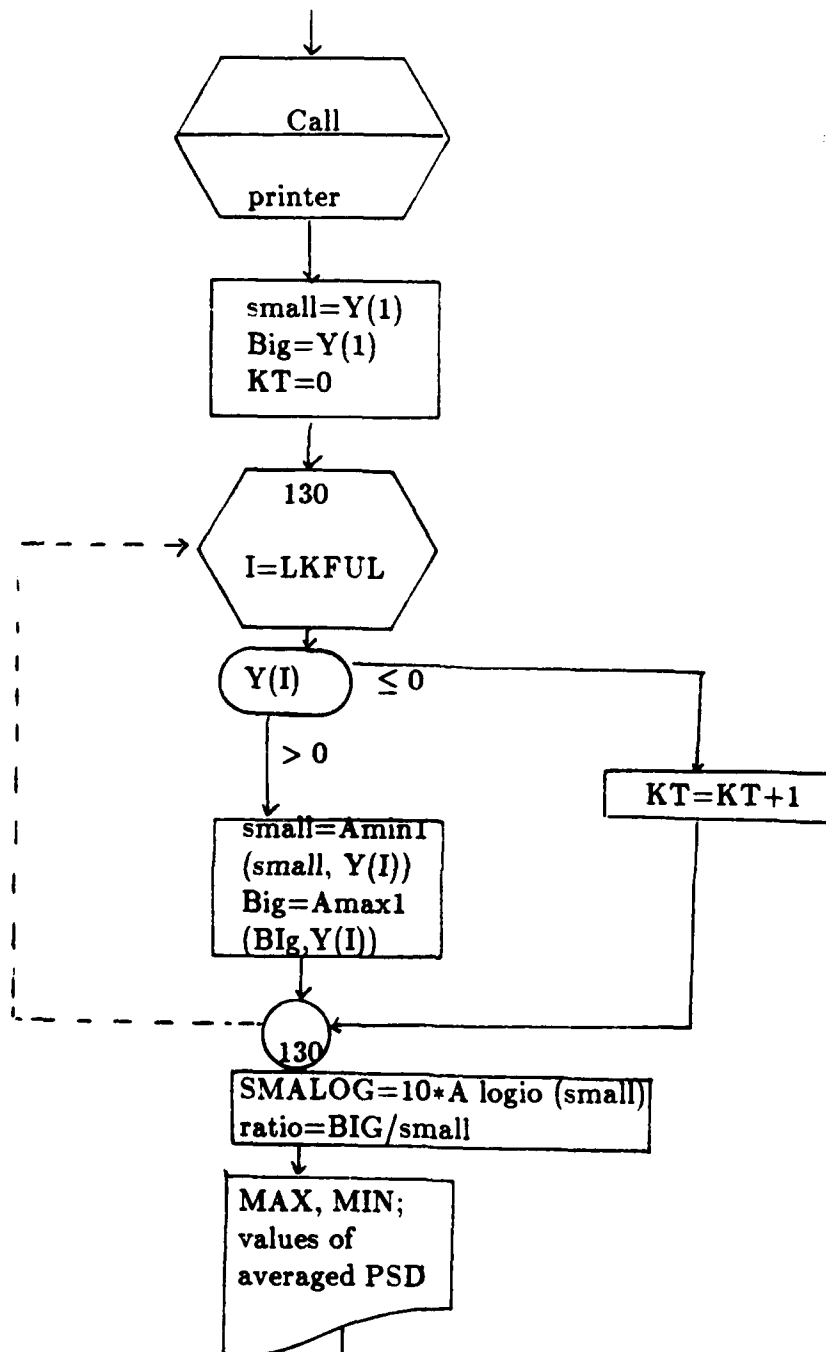


Figure 2 SPECTRA continued

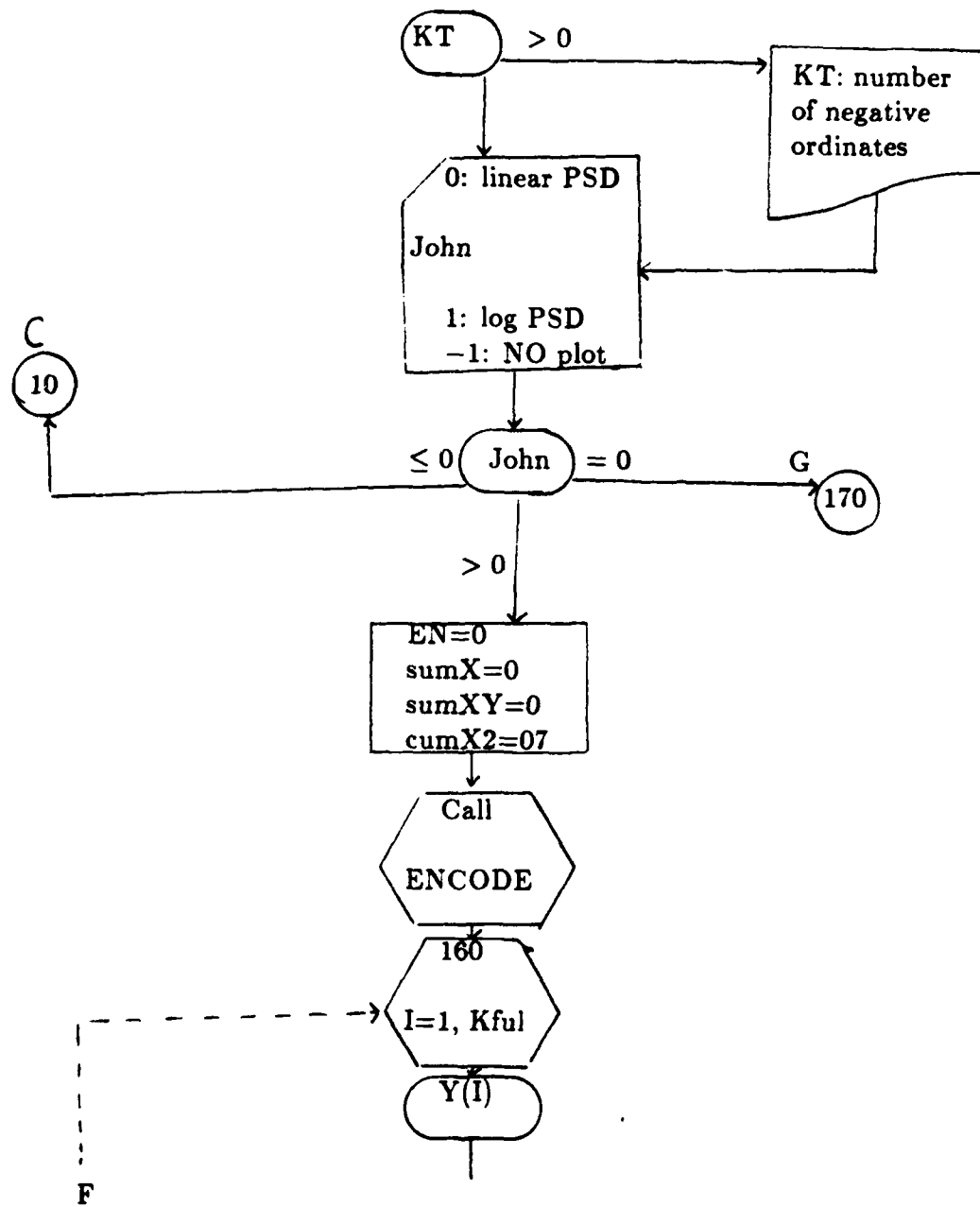


Figure 2 SPECTRA continued

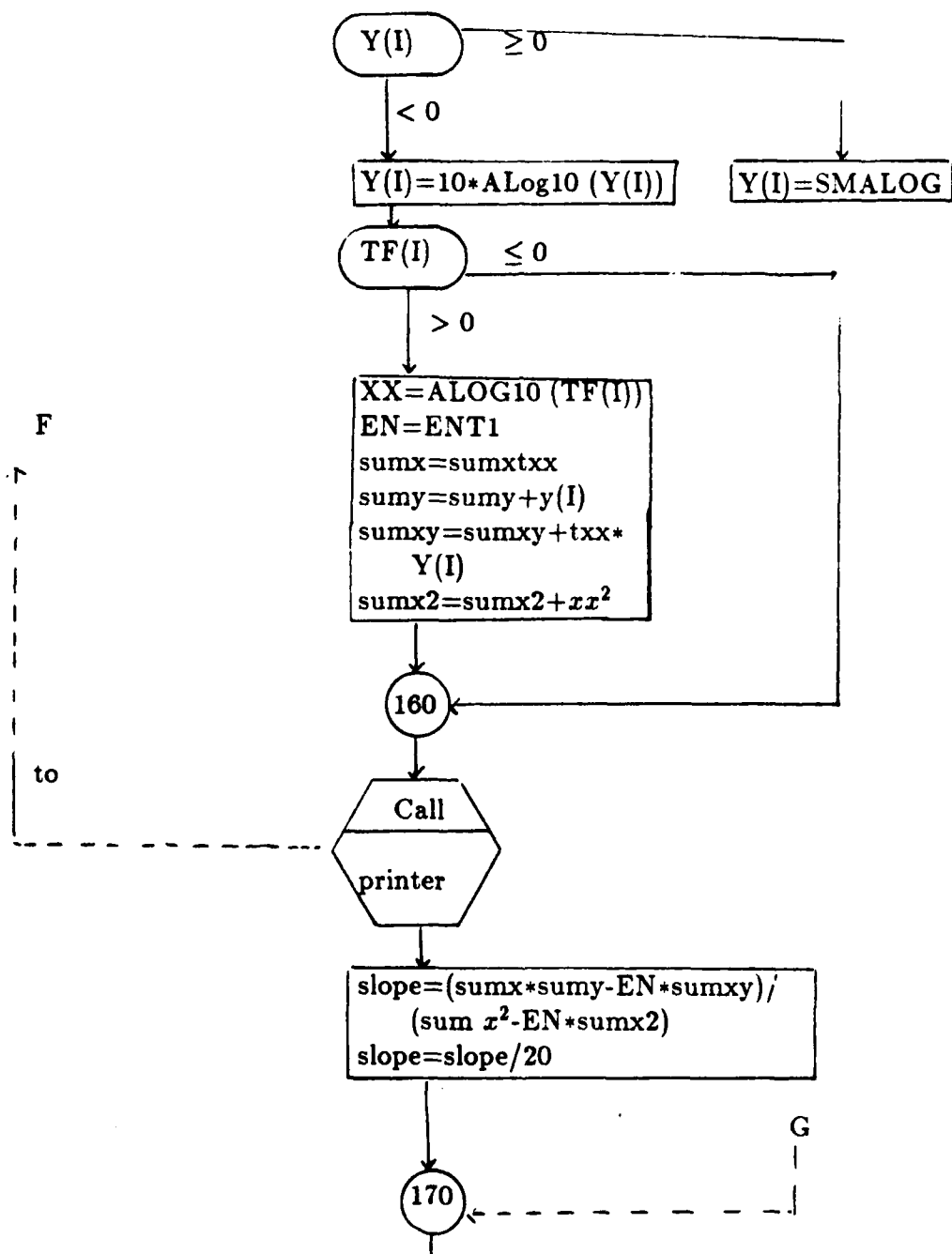


Figure 2 SPECTRA continued

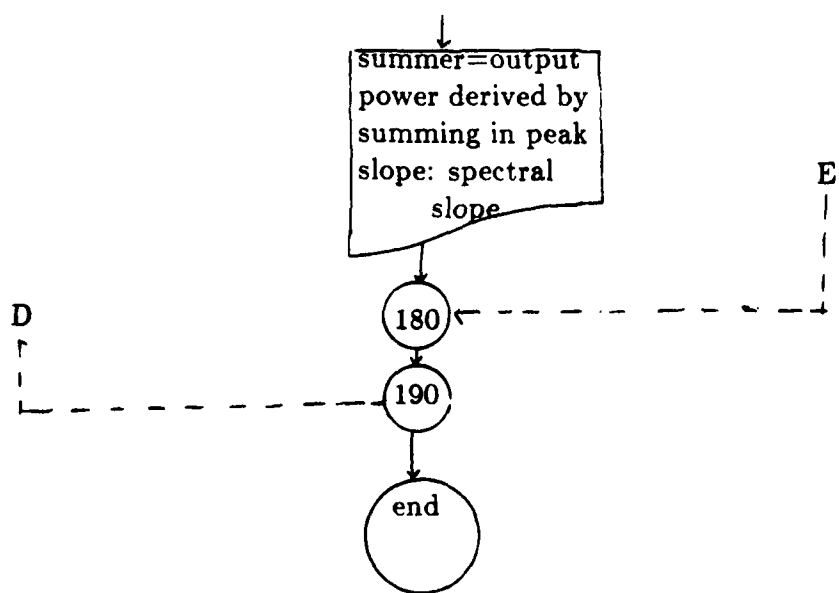


Figure 2 SPECTRA continued

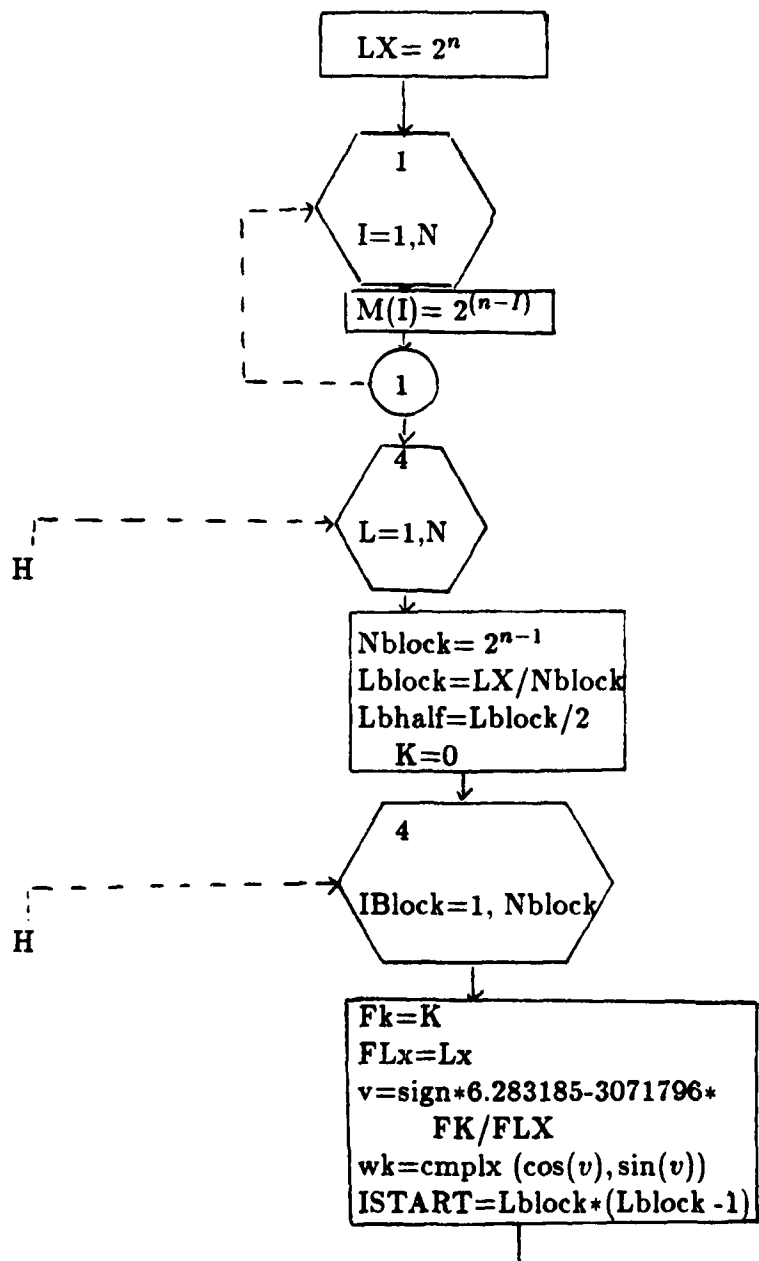


Figure 3 Detailed flow chart of NLOGN.  
showing all statements in the program



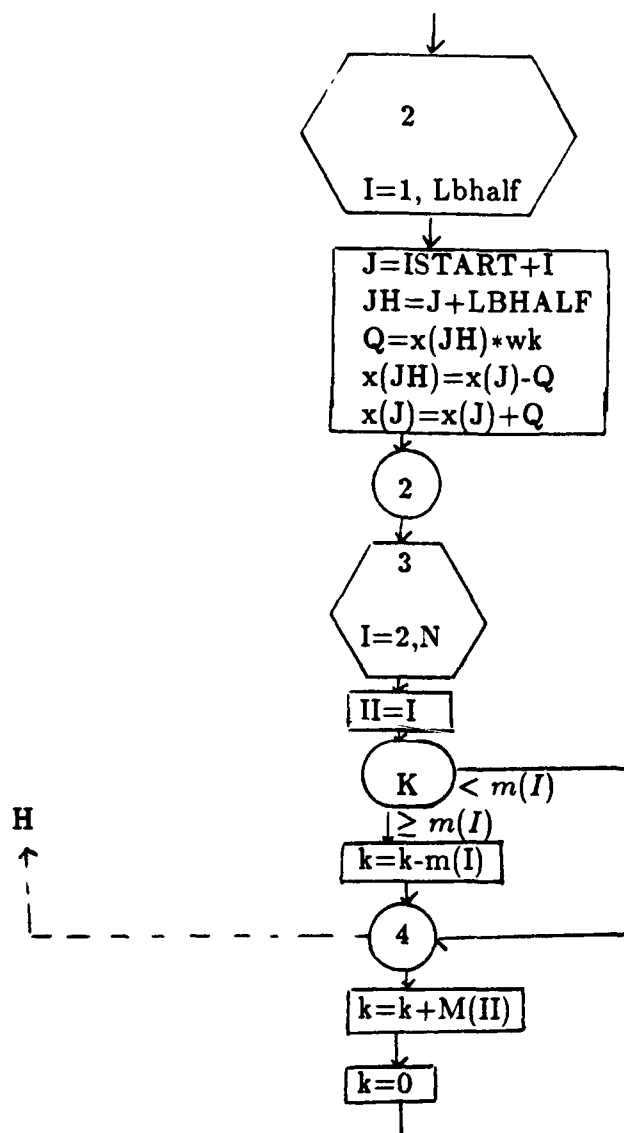


Figure 3 NLOGN continued

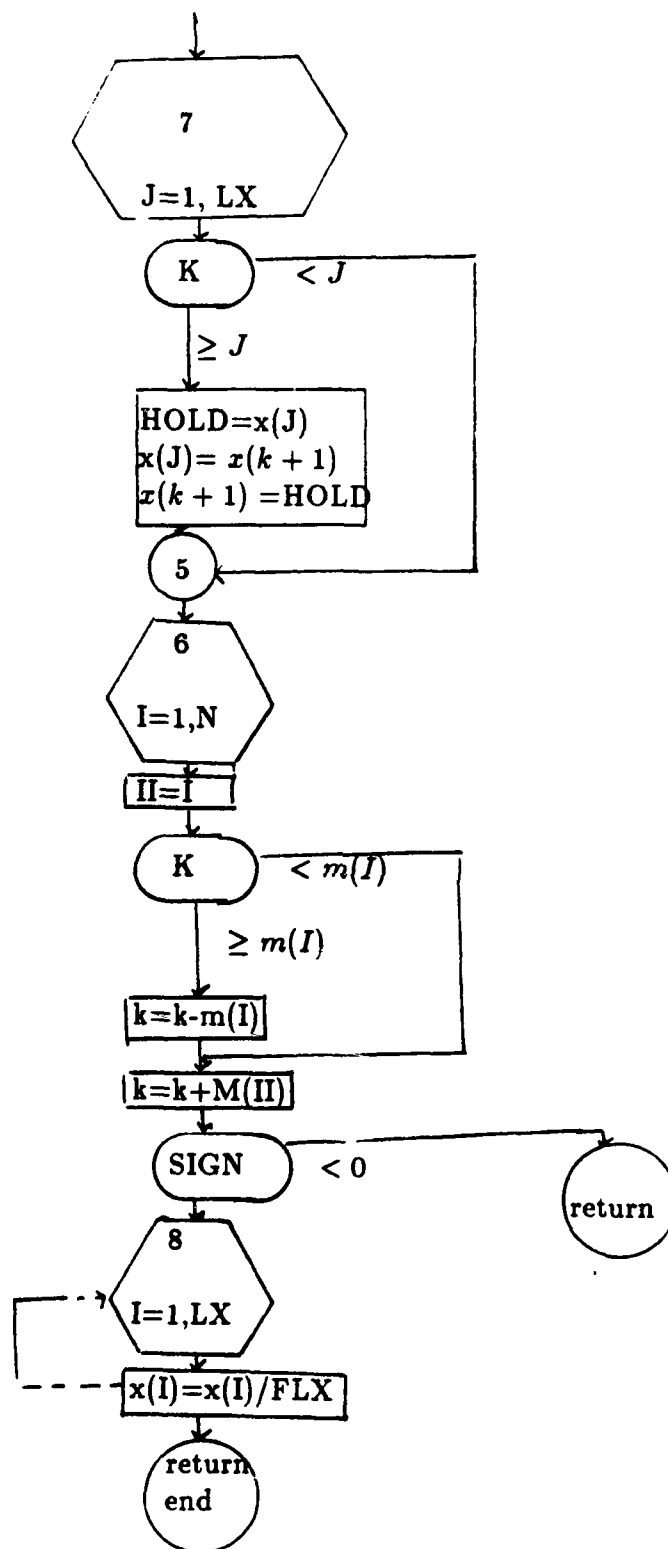


Figure 3 NGLOGN continued

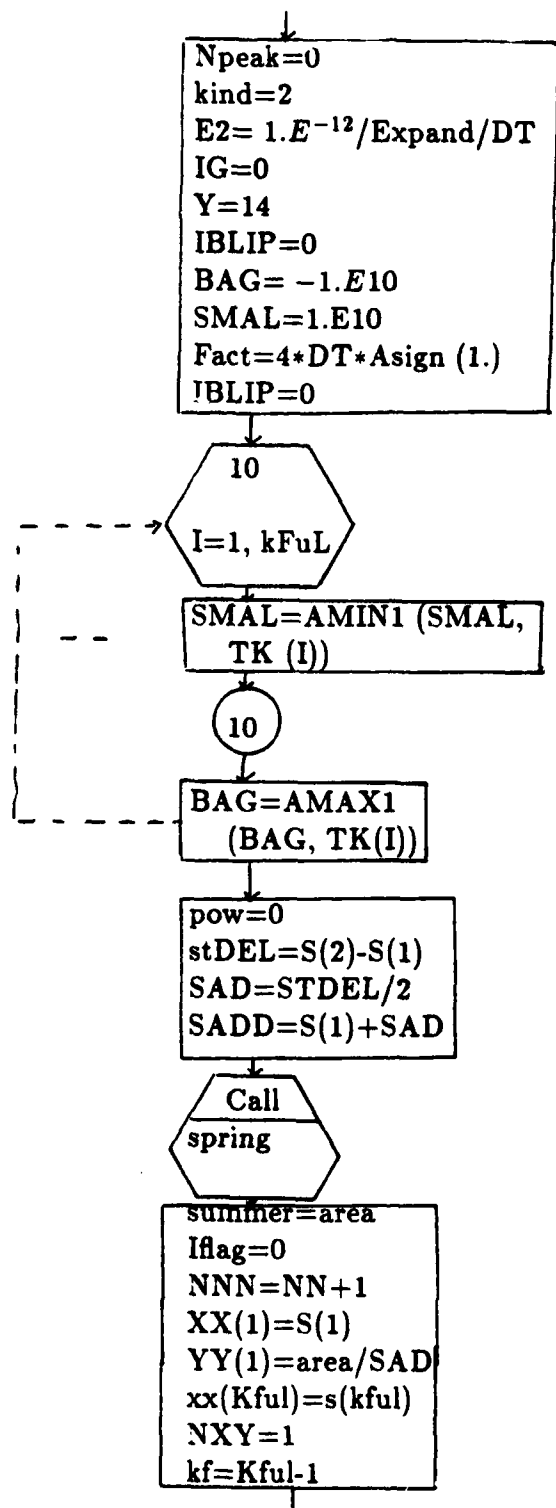


Figure 4: Detailed flow chart of PEAK.  
showing all statements in the program

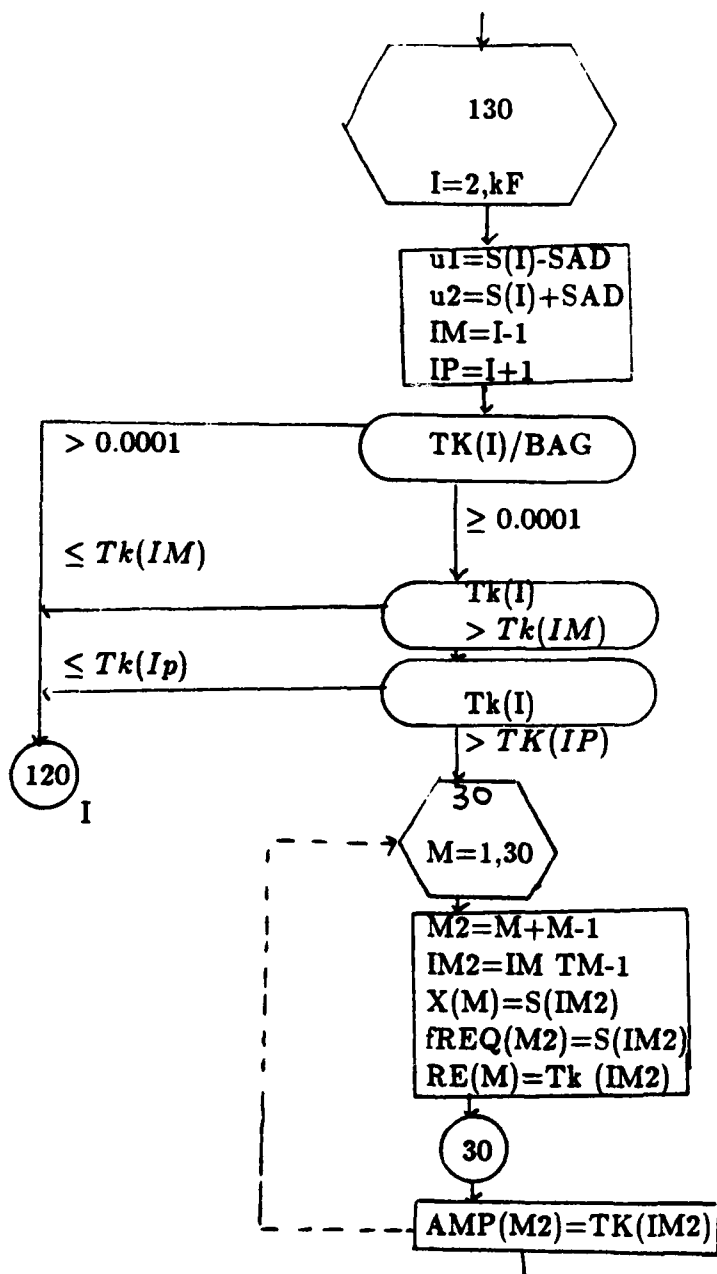


Figure 4 Peak continued

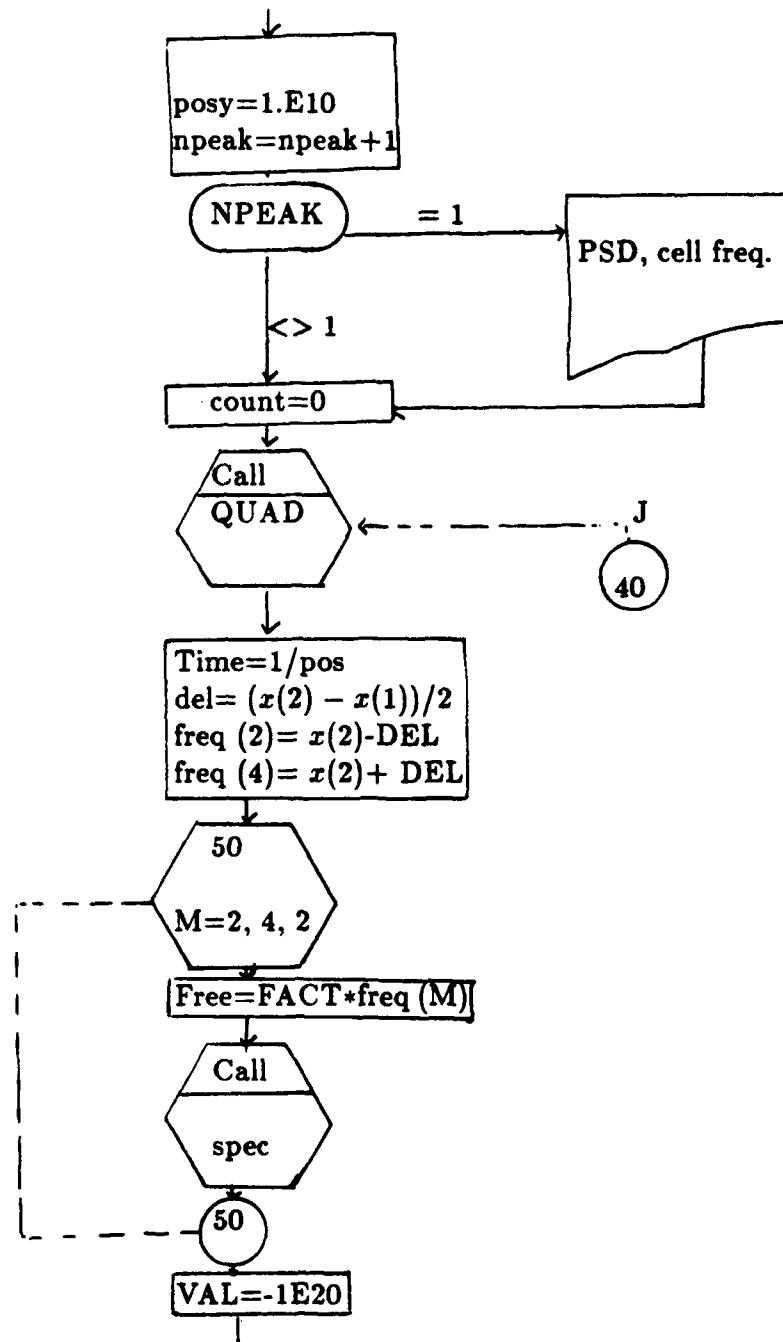


Figure 5: PEAK continued

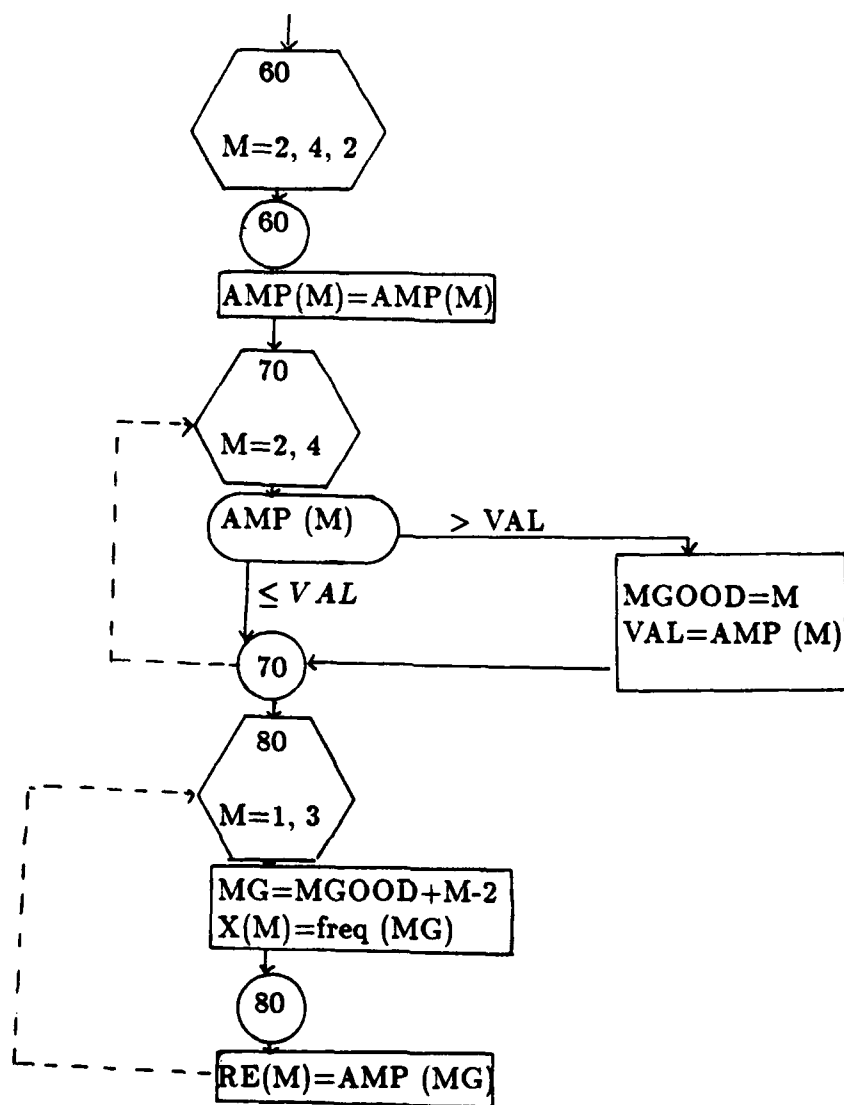


Figure 5 PEAK continued

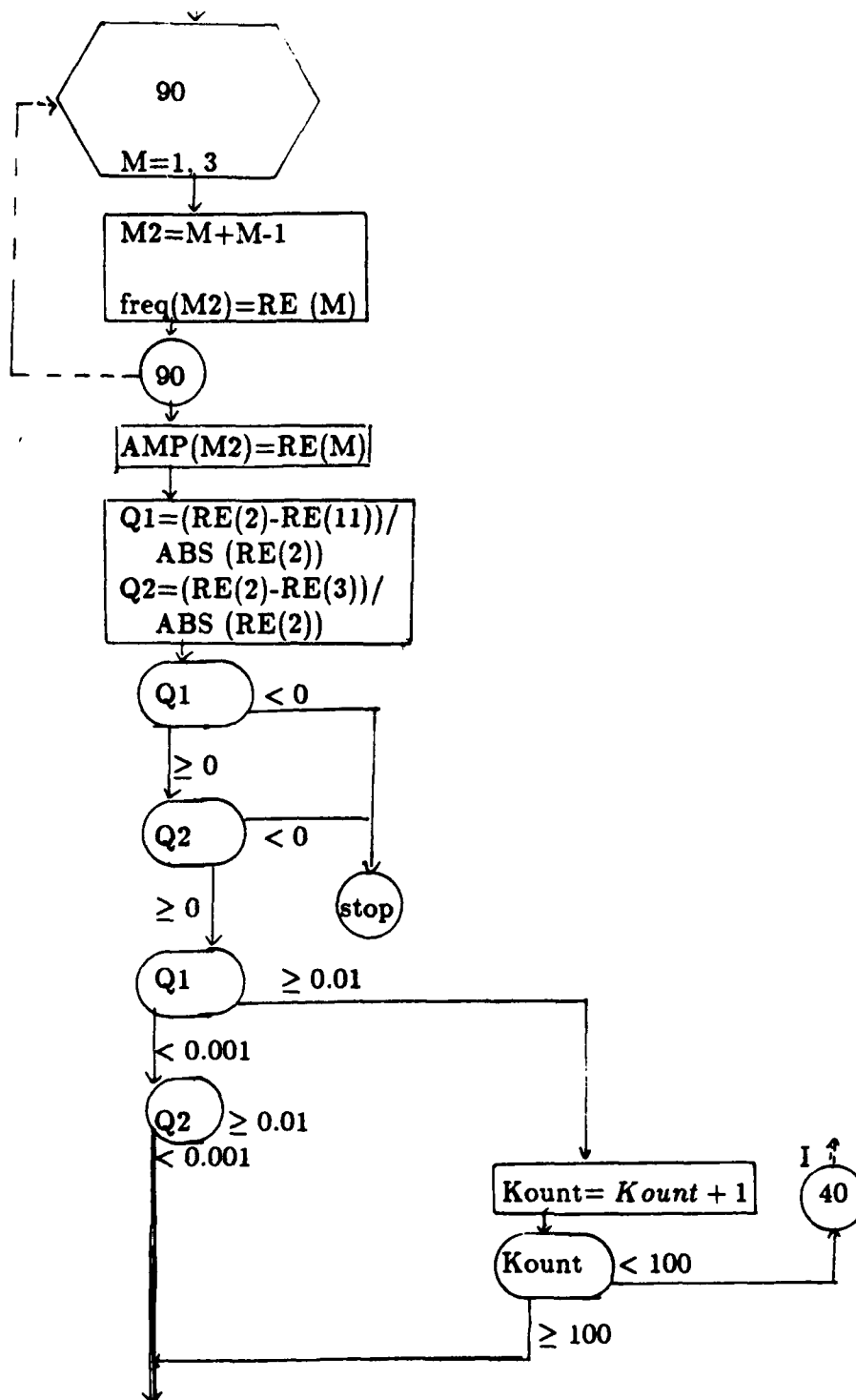


Figure 5: Peak continued

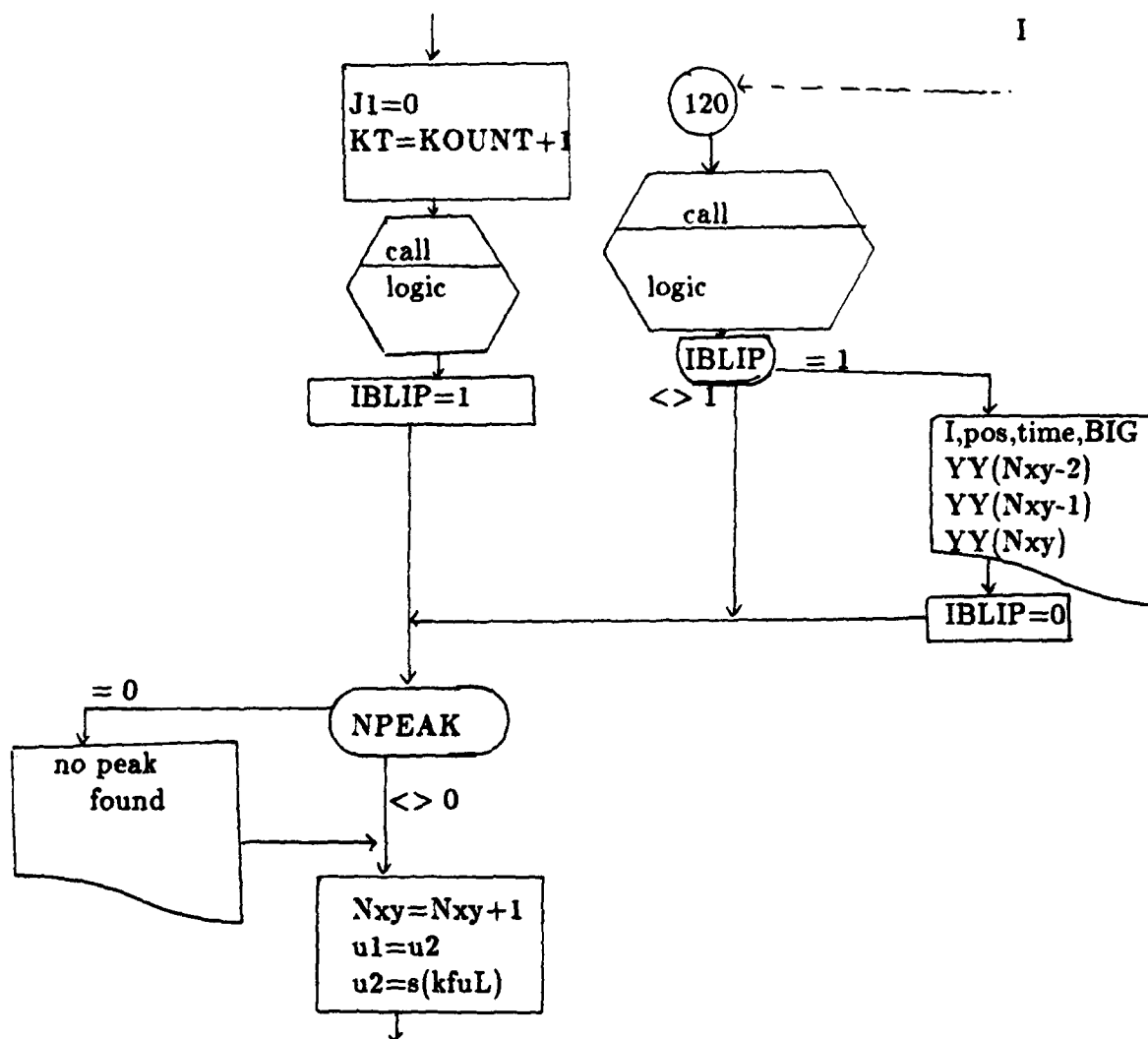


Figure 5 PEAK continued



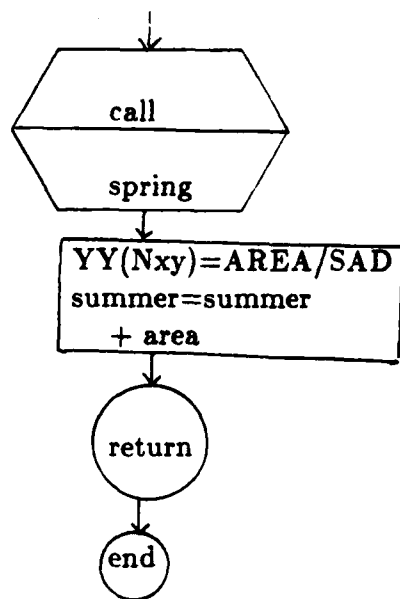


Figure 5 PEAK continued

Implementing Vortex II

by

Mary Ann Leekley

## ABSTRACT

The project goal was to fashion a computer operating system to replace an already existing system. This required tailoring a generic operating system to suit the particular situation. The original system and its replacement are discussed. The manner in which this new operating system was brought up is described in detail and the post-sysgen modification are described. Software written specifically for this new system is reviewed. Finally, the existing problems and future possibilities are noted.

## 1. PROJECT ORIGINS

### 1.1 SYSTEM HISTORY

The Air Force Geophysics Laboratory (AFGL) and Boston College started a joint project to study the earth's magnetic field and how it is affected by the solar wind. They set up a network of seven data acquisition stations with magnetometers across the continental U.S. which collected the data continuously. The network fed this data via Western Union lines to AFGL, Hanscom Air Force Base, Bedford, MA. A computer system was purchased to gather and process this raw data at Hanscom. The network's data was archived by this computer system and written onto seven-track tapes. This process was carried out in real time but not continuously. Between data reads and writes and the polling of the network, other programs were run analyzing these data. Of particular interest to the scientists that used the system were graphs and plots of their analysis of the magnetic field's activity.

### 1.2 OLD SYSTEM HARDWARE

The original system was purchased in 1973. Its CPU was a Varian V72 minicomputer with 32K memory (hereafter called the V72). The V72 was based on a three register, 16 bit word architecture. It was one of the Varian 620 series.

A Varian Diablo disc with 2 megabytes of memory was the main reason that the system was replaced. Sperry, which had bought part of Varian, had discontinued support of this type of disc. The scientist-users also needed more storage space for their programs. This Diablo held the operating system, the user programs, and the data for archiving on tape.

A Varian Statos 31 printer-plotter was not used. It had an address and a logical unit name in the operating system but it was disliked by the users. It needed repair but it was not clear how much would be necessary for the Statos to function properly.

A Tektronix 4040-14 CRT with a 4631 hard copy unit was heavily used. It was the opcom device and the output device. The plots and graphs copied off the Tektronix screen were reproduced on light-sensitive paper that darkened with time. Output was photocopied to get a permanent paper copy.

A Xerox Versatec 2030A printer-plotter was attached to a C-Tex controller. The C-Texcontroller could switch the copy of the Tektronix screen to the Versatec instead of the hard copy unit. The Versatec could not be addressed directly by the V72.

Three Varian seven-track tape drives were on the same controller. Five years of data were archived using these drives.

The Data Decoder Unit (built by UCLA, rebuilt by Dave Knecht and a programmer, Charlie Cantor) manipulated the stream of bits coming in from the Western Union lines and passed the words onto the V72.

The AWS was the device built by Dave Knecht to provide information from the network to the American Weather Service.

### 1.3 OLD SYSTEM SOFTWARE

The Varian Omnitask Real-Time Executive (Vortex) I, Version D, was the operating system. It ran user programs as either "foreground", i.e. real-time, or "background" which ran as the processor was available. The major real-time program was the data collection and archiving program which was written by the systems programmer at that time, Charlie Cantor. The system supported an assembler and a Fortran compiler. The user programs that had accumulated were written mostly in Fortran. In some cases the source code was missing. The old system's and the new system's resemblance in hardware and software were not purely coincidental. The new system had to look like the old in order to be able to run the old programs. Version D did not have many of the features and utilities that the more recent versions did. It did not have its own input-output utility for moving files. It also did not have a backup utility for loading and dumping individual files or partitions. These had been written by the users. There was supposed to be a spooler but it was not clear that it was really necessary given the way the system was used.

The old system was multi-user in a very limited sense. The network continually fed data to the CPU (students came in on weekends to supply fresh tapes). In the meantime, a single user could use the background to write, debug, and run his programs. The system was in constant use by the network and archiving program. It was not heavily used

otherwise.

## 1.4 NEW SYSTEM HARDWARE

It is important to note that the original system was completely satisfactory to its users. The only reason that it was being replaced was the cessation of support for the Diablo disc. They bought a system that was not very different from the old one because they wanted the same system they had before. Sperry Univac had bought the mini-computer operations of Varian, hence the change in names.

The hardware components of the new system were as follows:

A Sperry V77 CPU had the same basic architecture of 3 registers, 16 bit words but with 5 more registers available to the user. It included a 1024 byte cache and a floating point processor (FPP). The writable control store (WCS) was a separate board which could be used to program new instructions in microcode. Both the floating point processor and the writable control store were necessary to support the most recent version of Fortran (Fortran IV) offered by Sperry. The V77 had a 256K core. One of its features was an Automatic Boot Loader. It took five key strokes to boot the system on the V77, provided there was a viable system to be booted.

Two Sperry nine-track tape drives on one controller had 800 BPI and 1600 BPI capacity.

A Sperry Locust disc was a Winchester disc with 110 megabyte memory.

A Digital Decprinter I interfaced with the CPU via a teletype controller made by Sperry called a Universal Asynchronous Controller (UASC).

A Sperry UTS 10 terminal was purchased sans buffer so some of the keys' functions were not available to the user i.e the up-arrow. Some unresolved differences between the Vortex operating system and the newer hardware will be discussed later under Software Modifications Chapter 4

In the original specifications, all of the old system except the V72 CPU was to be included in the new system

## 1.5 NEW SYSTEM SOFTWARE

The operating system provided by Sperry was a Vortex II, 8R0. It had several useful utilities including an I/O utility and the backup utility. Vortex II utility, Miutil, supported the WCS which in turn supported the Fortran IV Compiler. The Dasmr assembler used all eight but indexing was still limited to two registers and there was no compare instruction. The set was smaller than the IBM 360's but one could eventually get the same results.

The V77 and Vortex II could have supported Cobol, Pascal, a database manager, etc. However, the primary users were intent on recreating the original system so the Fortran IV compiler and the Dasmr assembler were the sole inhabitants.

## 2. SPECIFICATION CHANGES

Changes in the project's funding caused a major change in specifications. The network of magnetometers was closed down in December 1983. The hardware that would have attached this network to the V77 was removed. The DDU and AWS became irrelevant. The major user program, the archiver, was no longer used. The real-time programming capability of Vortex had become useless. The main reason to replace the original system with a duplicate had disappeared.

In the original specifications, the Diablo disc was to hold the operating system. The Locust disc was to be the second disc. The old system continued to run with the seven-track drives, Statos, Tektronix, Diablo, and V72 while the new system was being brought up with everything else. When the new system was stable, everything but the V72 was to be brought over. However, it was difficult to see why the unsupported Diablo should be in the new system at all, much less have the operating system residing on it. It also created a major problem in system generation for the new system. The system generation program (Sysgen) expected the system disc's BIC (Buffer Interlace Controller) to be at a certain address. The Locust disc, because of its original secondary status, was not at that address. The Locust was rewired in order to put its BIC at the system disc's address and the users of the system agreed that the Diablo was not to be part of the new system. Ironically, during January, February, and March of 1984, the new Locust failed several times while

the old Diablo continued to rattle along.

In May, 1984, a meeting with Dave Knecht revealed a major change in the timetable for the new system's development. Instead of switching the remaining components (the Tektronix, C-TeX, 7-tracks, and Statos) when the new system was ready, Dr. Knecht had decided to wait until he was finished processing the data. This would take until September 1984 at least. This state of affairs is discussed further in Section 6.2.

### 3. SYSTEM GENERATION

#### 3.1 TOOLS FOR SYSTEM GENERATION

Sperry provided a tape entitled "System Generation Library". After typing in a bootstrap program at the console, the system generation program was loaded from the tape. The program (hereafter called Sysgen) used input from the console and/or tapes to generate an operating system and its library. The first thing that Sysgen needed was the location of the library tape (the Sysgen tape), the location of the directives tape, the alternate library (Section 4), the location of the system disc, and the location of the console. Unfortunately, the purchasers of the new system did not know that the output of a system generation can be very interesting. The UTS 10 did not have a hard copy unit. The Decprinter was unsuitable to be the Sysgen console as input from the console was required by Sysgen. So the listing of the modules and their addresses scrolled up the UTS screen, suitable for speed readers.

#### 3.2 DIRECTIVE CATEGORIES

The Sysgen program required a set of directives as input. Much of the effort to custom fit the operating system went into the directives. These were the parameters of the operating system and were divided into subsets as follows:

1. CPU type
2. clock intervals
3. peripheral equipment specifications (with and without controller tables).



4. priority interrupt module definitions.
5. logical unit assignments.
6. disc partition specifications.
7. memory configuration.
8. virtual nucleus overlay tasks.
9. external definitions.
10. library and nucleus modifications.
11. end directive.

These directives could be input either line by tedious line or the Sysgen program processed a tape of Ascii 40 word records. Sperry provided a tape entitled "Sysgen Helper". This tape loaded a stand-alone program so it was possible to create a tape of directives when there was no operating system.

Sysgen Helper was interactive. The correct answers that pacified Sysgen Helper had to come from manuals, the system memo, or Sysgen Helper's defaults. The system memo was the document describing the system's physical configuration. It was written for the customer engineer and considerable information could be gleaned from it. For the tyro, Sysgen Helper was indispensable. It had default answers for the bemused programmer who had not found the pertinent paragraph in the manuals yet. Some of these default answers were a little strange but would generate a system. Sysgen Helper could either create a tape from scratch or edit an existing one. The latter was useful when Vortex had crashed irrevocably.

### 3.3 CPU TYPE

The first directive declared which type of CPU there was. The choice of some modules depended on this number, particularly which of five versions of the general I/O handler was appropriate. Because the CPU was a V77 with a FPP and a WCS, it was called a type 5. The clock intervals depended on the CPU model. The basic clock interval and free-running counter increment were both hard wired. The user interrupt interval default

value was 20 milliseconds. This default value was left unchanged.

### 3.4 EQUIPMENT

The equipment directive had the device name, drivers, number of units, its BIC, and the number of retries. The system memo gave the device address of the peripherals, and the number of units.

A Buffer Interlace Controller was an I/O controller that allowed the peripherals to transfer data in a direct memory mode. It copied the initial and final addresses of the words to be transferred into its own registers. The task that needed to transfer the data was suspended until the task's BIC was through or until its timeout was over. How many BICs were permitted was not altogether clear. The BIC manual (Buffer Interlace Controller, p.1-1) said a maximum of four and the Programmer Reference (Vortex II Operating System Programmer Reference, p. 14-34) said a maximum of fifteen while the System Generation Guide (Vortex II System Generation User Guide, p.8-7) had a table of sixteen possible BIC interrupt addresses. At any rate, the system memo included nine, several of which became extraneous after the removal of magnetometer network.

The number of retries pertained only to hardware devices and the default number of twenty from Sysgen Helper was left in place.

Sysgen Helper and Sysgen associated certain addresses with certain devices. The difficulty with the Locust disc location has already been mentioned. Pseudo devices, i.e. the Spooler (SDOA), the memory dump utility (RMOG), and the file maintenance drive (FMOA) had their place in the equipment specifications without address, BIC, or retry.

The name of the equipment really meant the name of the nucleus module that was the I/O driver. Associated with these were the controller tables that held the storage for the I/O driver. With Sperry-Varian equipment finding the correct name was trivial except for the UASC that interfaced with the Decprinter. Because the USAC required a teletype module the Decprinter did not operate correctly. The solution to this is discussed in Section 3. The Tektronix used a UASC for its interface both on the old system and the new. The Versatec went into the equipment directives disguised as a Varian printer. This

is discussed in Section 5.

The number of units was a straightforward question answered by the system memo except in the case of the nine-track tapes. According to the system memo and the human eye there were two tape drives. However, the second tape drive (MT01) was unreachable during Sysgen, necessitating a single tape drive system generation. Worse, when the system was up, MT01 was still unresponsive. The customer engineer had wired it to the last location on the daisy chain MT03. The Sysgen program failed using the address MT03 unless the equipment directive claimed the existence of four units. Sysgen could swallow this lie and MT03 could be utilized. After some discussion, the C.E. persuaded himself that wiring MT03 to MT01 was a good idea. Thereafter system generation was done using both tape drives.

There was a set of directives pertaining to equipment without controller tables. The writable control store and the intertask communication module (ITOE) made their appearance here. This system had a large WCS with four "pages". Each page was equal to 1024 48 bit words. Each of these pages was a "unit".

### 3.5 PRIORITY INTERRUPT MODULES

Priority Interrupt Modules handled the interrupts. The modules for native Varian and Sperry devices used VSIOC, the common I/O handler for Vortex I and II. There had to be a PIM directive for every controller and its BIC. The directive required the name of the handler of the interrupt, the interrupt line number, the event word, and options. If the module was home-made rather than Varian-Sperry, one of the options had to be declared. The two devices that weren't Varian-Sperry were, for different reasons, given the names of regular Varian-Sperry modules. The Decprinter was controlled by a UASC using a teletype driver (VSTVA). The Versatec printer-plotter had a controller built by Versatec to resemble a Varian medium speed printer. According to the manual supplied by Versatec there was supposed to be an interrupt coming in from the Versatec. As far as could be determined by software experiments, Vortex II was not seeing any interrupt coming from the Versatec printer-plotter. So that device got an interrupt handler that never saw an interrupt. CRT's and TTY's had two event words which record interrupts and status, one

for output and one for input. Printers and other devices had one. A home-made interrupt handler had two choices, either to use VSIOC or be a directly connected handler. The VSIOC of Vortex II handled the Varian-Sperry I/O macros, i.e. write, read, etc. In order to use these macros, the names of a Varian-Sperry module was used. To maintain the deceit, the PIM and the driver for the Versatec had the names appropriate for a Varian printer.

### 3.6 LOGICAL UNIT ASSIGNMENTS

Logical unit assignments set up the connection between names and/or numbers for device addresses. These were divided into reassignable (by the user) and nonreassignable. A number of these could be left to Sysgen Helper's defaults.

The spooling system also had its own set of LUNs. All of the devices including each of the partitions needed LUNs. The new system was supposed to be able to run the user programs from the old system. Since many of the user programs used the LUNs, the new set of LUNs had to include almost all of the old names.

### 3.7 PARTITION

Partition definitions was a place where the Sysgen Helper behaved rather mysteriously. It was not clear, at first, how big to make the partitions. By relying on Sysgen Helper's default values, a system could be generated. However, when the time came to load the core, foreground, and background libraries, the process would fail due to insufficient space. Only a midget system could be built using Sysgen Helper's default partition sizes. The partition sizes were determined from the Sysgen Helper example and advice from another Vortex user.

Partition definition also included the storage key. Some partitions had to have certain Ascii letters as keys to suit Vortex conventions. Two more partitions were given keys to provide some security.

Vortex II managed a maximum of twenty partitions on the system disc. More partitions could be requested in the directives, but attempting to access a twenty-first partition would

generate an I/O error. The users' request for a larger number of smaller partitions could not be met.

The disc formatting program, which was run before anything else, did the track analysis and found alternates for bad tracks. Apparently, a large margin of error was allotted for.

### **3.8 MEMORY DEFINITION**

The memory directive determined the last address of the nucleus, the size of the foreground common area, and the total physical memory size. It also decided whether the system would use the extended register system and the long task identification block. It gave the number of 512 word pages allotted for storage of virtual nucleus overlay tasks and the last page number allowed for the virtual nucleus overlay tasks. This directive was written with the help of the system memo, Sysgen Helper, and Tom Rogers (Tom Rogers is a Vortex programmer in the Hartford, Conn. branch of Sperry).

### **3.9 VIRTUAL NUCLEUS OVERLAY TASKS**

The list of modules that were to be swapped in and out of the overlay portion of core included all the I/O drivers and the pseudo-drivers.

### **3.10 EXTERNAL DEFINITIONS**

External definitions made changes to system generation loader tables for special conditions. In the new operating system, external definitions declared the existence of the floating point processor and the intertask communication module. The Versatec's BIC was at a non-standard address. The length of the Versatec printer-plotter buffer had to be declared. The PATCH area, which was reserved for the PATCH utility at the bottom of the Vortex II nucleus area, was defined to be 300 words long, a generous guess. The WCS requirements were set here including another mention of its correct address.

### 3.11 MODIFICATIONS TO NUCLEUS AND LIBRARY

The modules VSRERF, VSRERS, and VSRER8, conflicted if they were all present. Since they had a direct bearing on the type of Fortran supported, the choice was important. The Sysgen manual was less than enlightening. After some experimentation using the ATP tape two arrangements were found to be satisfactory (Customer Approval Tape had a set of object decks to test the DASMR assembler and Fortran on a fledging system). If the Fortran that came with the Sysgen library was desired, the VSRERF module was deleted. If Fortran IV was required, both VSRERF and VSRERS were deleted.

Besides deletions, replacements and additions were included in this section of directives. In order to sneak the Versatec driver in, disguised as VZLPB, a replacement directive was included both for that driver and its accompanying control table.

### 3.12 END DIRECTIVE

The end directive makes the choice of whether the entire operating system was to be constructed or a "nucleus" only. The former, "standard generation", meant that the libraries, both the operating system's and the users' had to be reloaded. A nucleus-only generation "relinked" the libraries already existing. This was useful when trying out new, improved drivers. If any changes were made to the partition sizes or if the location of the PATCH area was changed, the nucleus-only generation would flounder.

## 4. SOFTWARE MODIFICATIONS

### 4.1 PATCH

The Vortex II operating system had a utility, PATCH, to modify code in the system's libraries. This was a very powerful (and occasionally dangerous) tool for putting changes into a running system. The code of the system itself could be modified or a jump could be made to the PATCH area where new code could be placed. The changes could be made temporarily and tested. They could be added to the file of permanent patches (PDLOG) which were applied every time the system was booted by PITCHIM. PATCH lent itself to small changes and experiments.

## 4.2 UTS BACK ARROW

One of the more annoying things about the UTS 10 was its inability to use the back arrow. Any backspacing was futile. The code for the teletype module, VSTYA, was released in 1974. The more recent assembler manual's octal equivalent for a back arrow did not match the VSTYA's equivalent. After testing the assembler manual's version, the change was made permanently to VSTYA. (See Appendix A, the AFGL Vortex II User Guide, Standard Sysgen for the exact code.)

## 4.3 DECPRINTER

When the Decprinter was assigned to be the output device, the machine printed over lines it had already done and occasionally printed far to the right of an 80 char line. The CPU was apparently transmitting the characters correctly but too fast for the Decprinter. The speed of transmission was reduced from 9600 to 2400 baud without improvement in the output.

The Decprinter did not send an interrupt to the CPU. So code was written to give pause to the CPU on every output to the Decprinter. A jump from the VSTYA output code went to new code in the PATCH area. If the device (VSTYA managed both the UTS and the Decprinter) was the Decprinter, then VSTYA called the Delay macro for the minimum amount of time. After the delay was over, the code jumped back to VSTYA. (See Appendix A, AFGL Vortex II User Guide Standard Sysgen, for exact code.)

## 4.4 CP77

When the first successful operating system came up, so did a development problem. The libraries included in the Sysgen tape did not include a text editor. The system on the V72 had an editor but only 7-track tapes while the V77 had the 9-track tapes. The transfer of the load module of the V72 editor via the AFGL mainframe (which handled 7 and 9 track tapes) seemed difficult without any guarantee of success.

The transfer of the V72 editor source code seemed like a better idea but would need editing itself. A helpful Vortex programmer, Bill Pierce (Bill Pierce works in the Sperry

branch in Hartford, Conn.) sent a tape of the CP77 program for installation. CP77 was a program that ran under Vortex II which supported multiple users, spooling, and, of immediate interest, an editor. This editor served well. There were serious problems with other parts of CP77, especially its spooler. The queuing module always hung up the system. The existence of the CP77 spooler conflicted with the Vortex spooler because they both needed the same LUNs. As the primary user of the system was completely uninterested in CP77's features and CP77 had no support (no source code), CP77 was removed. Dr. Knect wrote a program to dump the source code for the editor of the V72 onto the seven-track tape. This tape was converted to nine-track by another program on the AFGL mainframe and then loaded onto the V77. This editor became the editor for the new system. It required a minor addition in its code to make sure the control returned to the SO unit which was the UTS.

## 5. SPECIALIZED SOFTWARE

### 5.1 VERSATEC DRIVER

Versatec had built a controller board to resemble a Varian printer (620-77) and provided a number of manuals and a tape of the software package, Versaplot. The Controller Operating Manual contained code for I/O which was written for Vortex I rather than Vortex II. However, the code had important information, the function code numbers for exec and sens (sense the readiness) commands. These were nowhere to be found in the Sperry manuals.

By using the Varian-written driver (VZPLA and VZPLB) for a Varian printer as the basis for the Versatec driver, VSIOC, Vortex macros and other benefits were available. If the directly connected option had been followed, much of the code would be re-inventions. Therefore, the Versatec driver used the same conventions and entry names as VZLPB but was internally different.

VZLPB was the module driving a Varian printer with a companion controller table called CTLPOB which had the buffer. This buffer's length was enlarged to match the Versatec's buffer. (This also required a DEF directive )



VSIOC checked the macro's parameters before jumping to the specified driver. The only macros that would use the Versatec were write and func. Func (function) skipped lines and did form feeds for Varian printers. The Versatec was not going to deal with control characters such as the Fortran compiler produces so paper handling was done by func.

On entry to VZLPB (all the VSIOC drivers had a single entry point) the Versatec is tested for its readiness and then the Versatec buffer. If the Versatec was not ready, it is re-initialized, the status word of the request block is set, and control exits to the VSIOC error subroutine VSERR. VSIOC will try 20 more items. If the Versatec buffer is not ready, VZLPB puts itself into the Delay macro and when it wakes up, tests the condition again. This code was copied from the VZLPA which was clearer to read than VZLPB. After the Versatec and its buffer had been certified ready, the opcode was extracted from the request control block to decide whether to write or function.

The write macro of Vortex had several modes: system binary, Ascii, BCD, and unformatted. The system binary code was committed to plotting, Ascii to printing, and BCD to SSP (simultaneous printing and plotting). The last was not used by the Versaplot package. After deciding which mode was needed, control of the program jumped to code that executed commands that set the mode in the Versatec. The bookkeeping was set up to keep track of how many words had been sent to the Versatec, how many were to be done, and the pointers to the user buffer space and the VZLPB buffer space in CTLPOB. For both print and plot, data was output one byte per word, that byte being in the right hand side of the word. For the print the leftmost bit was stripped off so that the character became a teletype character. For the plot that bit was left alone. The words were output after testing (again) the readiness of the Versatec buffer.

The original VZLLPB expected an interrupt after outputting a character(s). If there was no sign of an interrupt in the event word of the request control block after the end of the delay macro, the error condition was set. According to the controller operating manual, there was supposed to be an interrupt but it was never found despite the obvious printing and plotting. So the test for the interrupt was dropped.

The Ascii control characters for carriage return and line feed were ignored by the Versatec because it had more than 128 characters in its ROM. A full Versatec buffer automatically printed or plotted. A line terminate could be executed in order to force a print or a plot of the buffer's contents.

VSFNR (another VSIOC subroutine) took a graceful exit after the status word had been set, returning control to VSIOC. There also existed VSBIC which set up the correct conditions for DMA transfer via the BIC of that controller. The original VZLPA and VZLPB code both used the BIC for output exclusively. The results of using the Versatec with the BIC were erratic. So the code using the BIC was dropped in exchange for the output character by character. Lacking a discernable interrupt from the Versatec it was necessary to use the "sen" (sense) command to check if the Versatec buffer was ready to accept data. Otherwise, data was lost with interesting results to the plots.

The fun macro called for form feeds (1.5 inches) or end of transmission (8 inches) and executed those commands and exited via VSFNR.

## 5.2 VERSAPLOT INTEGRATION

Versaplot package needed a set of Fortran callable subroutines ( dopen, dwrite, dread, dclose, dwait) to do its disc I/O. Disci2 was written in assembler to meet Versaplot's requirements. Vortex II's file manager (FMAIN) could not allocate file space on the fly. The easiest way around FMAIN's limitation was to create the files required by Versaplot ahead of time and leave them in place. There was a precedent for this; the Spool files and System Dump files had to be created prior to program execution. Disci2 set up the I/O macros read, write, etc. in the standard fashion with several exceptions. The Versaplot modules would call for a file to be closed and deleted but the Disci2 only closed, never deleted. A call to open would open an already existing file. A close and keep would close a file and put a pseudo-end-of-file on the next record. When that file was reopened and read, the pseudo-end-of-file condition was checked.

Versaplot also needed to send its buffers of rasters to the Versatec via Fortran-callable subroutines, MTX, MTSET, MWAIT. The MTX code was written specifically for the

AD-A194 244

MINICOMPUTER AND DATA ANALYSIS IN SUPPORT OF THE AFGL

272

(AIR FORCE GEOPHYSI.. (U) BOSTON UNIV MA

R B D'AGOSTINO ET AL. 22 OCT 84 AFGL-TR-84-0276

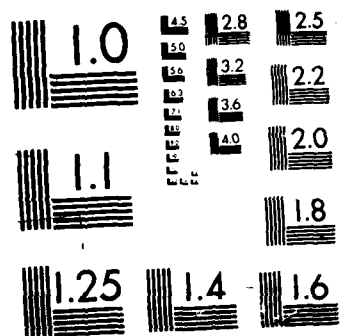
UNCLASSIFIED

F19628-82-K-0044

F/G 14/2

NL





Versatec driver and used the write, function, and delay macros.

The Versaplot modules themselves required some adjustment as the Vortex Fortran IV compiler required knowing about all the common areas in the main program and did not accept octal numbers as data, only hexadecimal or decimal.

## Chapter 6

### SYSTEM PRESENT AND FUTURE

#### 6.1 PRESENT

The new system is now as useable as the old system. The new system has more utilities (FMUTIL in particular) for the user and a more recent dialect of Fortran. For future development, several issues should be mentioned.

The Sysgen procedures cannot be simplified any further without either writing a new program to create complete SGL tapes or acquiring JSEdit which is a special editor for such tasks. The SMAIN utility deals only with the SLG nucleus and not with the library creation.

The Directives tapes have on them all the directives necessary for the completed new system. The directives pertaining to equipment still on the old system are commented out. There is no guarantee that the directives for the complete system are correct as they have not been tested.

The PATCH utility will not read in change directives from RMD files. This means the Sysgen has to be followed by typing in the new patches written for this particular system.

The DYSTEM utility, the system debugger, does not work. It will not allow access to the file containing the dump.

The 7-track tape drives, the Tektronix, and the C-tex have yet to be added to the new system. The tape drives should not be any problem except for their age. The C-tex and the Tektronix however could be tricky. There is no source from the old system on what modules and modifications were used to control them. There is a source code that uses "TEX" as a label name but the code is uncommented and obscure.

## 6.2 FUTURE

The system as it stands is a single-user system. Sperry-U-nivac does have multiple-user subsystem, CP77, which was tried once (See Section 4-4). That installation did not succeed due to lack of support. (In fact, the installation tape came in a brown paper wrapper.) Correct installation of CP77, which may need some additional hardware, would substantially increase the productiveness of the system.

On Hanscom AFB there are a number of computers and a local area network has been installed to connect them. This system could join this network and make the magnetometer data more accessible.

## BIBLIOGRAPHY

- Digital Equipment Corporation. *LA180 Decprinter I User's Manual, #EX-LA180-OP-009*. Maynard, Mass. 1977.
- Sperry Univac. *Assembly Language Programmer Reference. UP-8682, Rev. 1*. Irvine, Calif, 8/80.
- Sperry Univac. *Buffer Interlace Controller Model 7X-3103 (F3024-02) Operation and Service, UP-8626, Rev 1*. Irvine, Calif. 4/80.
- Sperry Univac. *Control Program-77, Time Sharing System Reference Manual, UP-8714*. Irvine, Calif. 5/79.
- Sperry Univac. *Control Program Time Sharing System User Guide, Up-8715*. Irvine, Calif. 5/79.
- Sperry Univac. *System Memo, JR-83-077-S.C. (V). REv A*. Irvine, Calif. 3/21/83.
- Sperry Univac. *V70 Series Architecture Reference Manual, UP-8634, Rev 1*. Irvine, Calif. 10/79.
- Sperry Univac. *Vortex Fortran IV Programmer Reference, UP-8671, Rev 1*. Irvine, Calif. 2/81.
- Sperry Univac. *Vortex II Acceptance Test Procedure, 92W0711-004H1*. Irvine, Calif. 8/80.
- Sperry Univac. *Vortex II Operating System 8R1 (SRD), 92W0711-004H1*. Irvine, Calif. 9/25/81.
- Sperry Univac. *Vortex II Operating System Programmer Reference, Up-8677, Rev 3*. Irvine, Calif. 4/81.
- Sperry Univac. *Vortex II Rev 8R0.0 9220711 004H0 (microfiche)*. Irvine, Calif.
- Sperry Univac. *Vortex II Operating System Programmer Reference, Up-8677, Rev 3*. Irvine, Calif. 4/81.

Sperry Univac. *Vortex II System Generation User Guide/Programmer Reference*, UP-9083, Rev 2. Irvine, Calif. 4/81.

Sperry Univac. *Vortex II Sysgen Helper User Guide*, UP-8924, Rev 1. Irvine, Calif. 6/81.

Versatec. *Bulletin No 312*. Santa Clara, Calif. 9/77.

Versatec. *Controller Operating Manual (C-620/M77)*. Santa Clara, Calif. 8/75.

Versatec. *Controller Operating Manual Model 235 (C-TeX5)*, ed 2, Santa Clara, Calif. 2/78.

Versatec. *Operation and Maintenance Manual 2030 Series*, ed. 2, Santa Clara, Calif. 1/78.

Versatec. *Versaplot-V07, Operational Design and Integration Manual*, 50028-90002, ed 1. Santa Clara, Calif. 9/77.

Versatec. *Versaplot Software Manual*, 50028-90001. Santa Clara, Calif. 12/76.



## Appendix A

### AFGL VORTEX II USER GUIDE

#### A.1 UTS CONTROL PAGE

In order for the UTS to work the control page has to be set up as follows:

DUPX (F)    BAUD(2400)    PARITY(M)    STOP(2)  
AUTONL(Y)    RET=NL(N)    I/F(C)    FRQ(60)  
CYCP(N)    ANSBAK( )    KK(Y)    ROLPG(Y)

#### A.2 HALT AND BOOT

In order to halt the system do the following;

1. Hit the blue key "Break".
2. Type "H" on the regular keyboard.

In order to boot the system using the Automatic Boot Loader (ABL) do the following:

1. Type "A" on the regular keyboard.
2. Type "0" which gives the contents of register 0. Then type "2" immediately after the contents of the register and hit return. You have changed the contents of register 0. If you want to make sure, type 0 again and put a period after the end of the contents to maintain the contents.
3. Type "B" on the regular keyboard.

There will be a pause. Then the UTS will announce that patching has been completed. It will then prompt for the date and the time. If there is no file for the Dsystem utility (Image0 in 190,Z) an error message will appear but is not fatal.

#### A.3 SYSTEM GENERATION

##### A.3.1 Types of Sysgen

System generation (Sysgen) is not difficult, merely tedious. There are at least two situations which require Sysgen: when a major change is made to the system, i.e., new equipment or changes in the nucleus, or when the disc has failed completely. There are two kinds of Sysgen, "Standard" and "Nucleus". The Standard Sysgen creates the whole operating system including the libraries, compilers, and patches. The Nucleus Sysgen creates the nucleus only and "relinks" foreground and background libraries. The procedure is the same for both except the Nucleus Sysgen is shorter. The Nucleus Sysgen is usually sufficient except when the partition sizes are changed, the patch area overwritten, or the disc reformatted.

### **A.3.2 Standard Sysgen Procedure**

In order to do a standard Sysgen, you need the following:

1. SLG tape.
2. Directives for Standard Sysgen tape.
3. CTLPOB object deck tape.
4. VZLPB object deck tape.
5. Fortran IV tape.
6. PATCH tape.
7. copy of the bootstrap program (p. 2-9 of System User Guide).

Turn the Decprinter "on-line". The UTS should be set to "caps only", using the blue key. Turn the key on the computer panel to "reset" and back to "on". Mount the SGL tape on MT00 (the one closest to the door) and the Directives tape on MT01 (the one next closest to the door). Key in the bootstrap program using the method outlined in System Generation User Manual p.2-9. The Sysgen program will be loaded and then prompts on the UTS, "I/O INTERROGATION". The following must be typed in at the UTS.

```
DIR,MT01A,022,010  
LIB,MT00A,022,010  
ALT.MT01A,022,010
```

LIS, TY001, 01

SYS, D00G, 020, 013

The backarrow on UTS does not work at this stage (it will after you finish putting the patches in). If you make an error, use the backslash and retype the line. When all five lines are typed correctly, Sysgen proceeds. Refer to System Generation User Manual p.6-2 for more detail on this part of Sysgen.

Sysgen then prompts for a header for the Directives. Respond with a carriage return and the Directives are read off the Directives tape. As soon as the partitioning shows on the UTS, remove the Directives tape and mount the CTLPOB object deck tape. Sysgen prompts "REPLACE CTLPOB READY". Respond by typing in "ALT". Sysgen prompts "READY." Respond with "LIB" and then replace CTLPOB tape with VZLPB object tape. Sysgen prompts "REPLACE VZLPB READY". Respond by typing in "ALT". Sysgen prompts "READY". Respond with "LIB" and then replace the VZLPB tape with the Fortran IV tape. Refer to Sysgen Generation Manual, p.6-49 for more detail on this part of Sysgen.

Sysgen lists the module addresses on the UTS screen and then the physical memory allocation. After pages Sysgen says "READY TO LOAD LIBRARY" and 2 EX and 2 BLD errors appear. Set the time and the date using the opcom commands. For more detail on this part of Sysgen see p.2-37 of the System Generation User Manual.

Now you load the system libraries from the SGL tape by assigning SI to M5 (MT00). Sysgen explains each job as it appears on the tape but the following is exactly what needs to be loaded from the tape. For more detail see p.2-38 of System Generation User Manual.

TABLE 1

## Library Jobs

JOB STREAM NAME	RUN	DESCRIPTION
BSCOM	yes	OM library
VSRERS	no	Fortran OM
VSRERF	no	Fortran OM
SFTOM	yes	Math OM(which forces sfile to FMUTIL)
ALCOM	no	Math OM
FPPOM	no	Math OM
FPAOM	no	Math OM
MUTIL	yes	utility for backup
ST. LIB.	yes	
FORTTRAN COM	no	
VSORT	yes	background vsort
VSORT	yes	foreground vsort
RPGIV	no	RPG compiler
V77-WCS	no	for V77-400
V70-WCS	no	for commercial firmware
VTNOM	no	VTAM
VDPOM	yes	Vortex Compatible Dataplot
MDPOM	no	MOS Compatible Dataplot
V77-600 parity	no	parity for 600
V77-800parity	yes	parity for 800
COMSY	yes	Comsy for WCS
800WCS	no	commercial firmware

Assign SI to M4 (MT01) and load the Fortran IV tape contents, which includes the WCS, the compiler, and the library. Keep assigning SI to M4 until the job tells you that the end of the job has been reached. For more detail on this tape see System Release Document (SRD) p.7.1.

Replace the Fortran IV tape with the PATCH tape. Type `"/FINI"` and then `";SCHED,PATCH,5,FL,F"`. After the PATCH prompt `Pt*`, type `".SLCT,M4"`. PATCH loads the patches to the system nucleus from the tape. When `"PATCH EXITING"` appears, rewind the tape, skip one file, type `"/FINI"`, type `";SCHED,PATCH,5,FL,F"` and finally `".SLCT,M4"`. PATCH loads the patches to the system libraries from the second file on the tape. For more detail on this, refer to the SRD p.A.1. Break, halt and reboot the system so that the patches are implemented by PITCHIM.

In order to get in the patches that make the UTS backarrow and the Decprinter to work, you must schedule PATCH again and after its prompt, type in the following exactly. If there are any modifications to the nucleus, i.e., addition of modules for the Tektronix, the Patch area will probably be changed. After doing the above addition of Patches by tape, look at the PATCH area by using the `.H` directive to see what the next open address is and change the 31157 in the following code to that address.

`.A`

```
31157,TAB,LDA+422,ERAX+6,JANZ,31157+14,TXA,JSRX,406
31157+10,1100,1,0,TAX,TBA,ROF,LDB+432,JMP,VSTYA+2037
VSTYA+2035,JMP,31157
```

`.E`

Break and boot the system again to add these patches.

Change the size of PO and SS to suit the size editor you want. Create SPOOL0 on SX,S for the Spooler to use. Load partitions from backup tapes using FMUTIL utility LOAD command with ALLNW so you don't overwrite modules the system has already put in.

### A.3.3 Nucleus Sysgen

In order to do a nucleus Sysgen you need the following:

1. SGL tape.
2. Directives for Nucleus Sysgen.

The procedure is the same as for the Standard through the input of the Directives. The system takes it from there and relinks the already existing foreground and background libraries. See p.4-7 of System Generation User Guide.

#### **A.4 VERSATEC AND VERSAPLOT**

The software package Versaplot has three different algorithms; mapped, linked, and sorted. They are mutually exclusive. The mapped algorithm has been implemented. If the decision is made to switch to linked or sorted, the integration has to be redone using the Integration Manual. The source code of Versaplot is called 'Xversa' and exists in its original state in D6 and on the backup tape for that partition. The following is a table of the modules for Versaplot. The source code is in D6 and the modules in OM except for RASM which is in BL. They are all backed up by FMUTIL partition dumps onto tape.

TABLE 1  
Versaplot Modules

SOURCE	OBJECT MODULE
pread	pread
pwrite	pwrite
irams	iram
lshift	lshift
disci2	dopen,dclose,dwait,dwrite,dread
mtx	mtxset,mtx,mwait
ppep2	pep2bd
invect	invect
cycler	cycler
irzero	irzero
spot	spot
deque	deque
queue	queue
newpen	newpen
factov	factov
wherev	wherev
offset	offset
symbov	symbov
axisv	axisv
linev	linev
scalev	scalev
setmsg	setmsg
versm	vers
vrinim	vrinit

vector	vector
incept	incept
grid	grid
tone	tone
curve	curve
plotsv	plotsv
plotv	plotv
SOURCE	OBJECT MODULE
rasm	rasm (a main program)

There must be a file called "PARM" of at least 50 sectors long in SX and "VECTR1" of at least 200 sectors long in SX. If the linked or sorted algorithm is implemented, then "VECTR2" and "VECTR3" must be added to SX. PARM and VECTR1 are backed up on tape.

The driver for the Versatec, VZLPB, and its accompanying table, CTLPOB, exist in the nucleus. The source code for both are in D6 and are backed up by tape.

See the Versaplot Software Manual on how to use the Versaplot package.

## A.5 LOCUST DISK

Before turning the disc off, type "/fini" on the UTS. Turn the protect file switch up (so the light is on) on the disc front before turning the power down (lights go off). When turning the disc back on, flip the power switch up, then flip the protect file switch down.

If the room temperature goes above 70 degrees F. the controller for the disc will probably misbehave. Do not panic, just wait for the room temperature to go down again and it will probably recover on its own.

If the disc has to be reformatted, see the SDR on Disc Formatting which is in the same looseleaf notebook as the System Generation User Manual. There are directions specific to the Locust and the Format tape.



## **A.6 DECPRINTER**

The Decprinter has several peculiarities due to the fact it is being run by teletype software. If you touch the keyboard of the UTS while there is output to the Decprinter, you will affect the output. On a long listing, the Decprinter will sometimes pause what seems like a very long time. Do not panic, it will continue.

### **A.6.1 Partitions**

The following is the table of partition, logical name(s), numbers, and keys (if any).

TABLE 1

## Locust Partitions

D00( 1)	CL	103 C
D00( 2)	FL	106 F
D00( 3)	BL	105 E
D00( 4)	OM	104 D
D00( 5)	CU	101 S
D00( 6)	SW	102
D00( 7)		116
D00( 8)	GO	9
D00( 9)	PO,SS	8,10
D00(10)	BI	6
D00(11)	BO	7
D00(12)	WZ	190 Z
D00(13)	D3	33 S
D00(14)	D0	30
D00(15)	D4	34S
D00(16)	SX	107 S
D00(17)	D1	31
D00(18)	D2	32
D00(19)	D5	35
D00(20)	D6,PL	36,50

END

DATE

FILMED

8-88

DTIC